CHAPTER 1 - INTRODUCTION TO FRAME RELAY TECHNOLOGY

Introduction

The map of the world for wide-area networks has been pretty much taken for granted. Local networks can, and do, change daily but WANs were defined by their stable borders and clearly delineated territories. This was a state of affairs not expected to change for a long time.

However, these days, there's a major shift taking place, on account of widespread economic and technological changes in public voice services and private data networking. It can be seen in the move to put private voice back onto the public telephone network while data networks retain their own T1 pipes. This may well result in the total *disintegration* of the integrated voice and data networks the industry toiled so hard to build.

For data, the shift away from asynchronous terminal-to-host configurations and toward distributed computing is relentlessly increasing traffic. And as peer-to-peer distributed applications become the norm, the demand for low-speed data communications will remain constant while the need for high-speed data will grow.

Today, with economics and technology favoring divergent voice and data networks, another generation of wide-area backbone equipment is emerging, one that emphasizes LAN interconnections. This new gear sports T1, fractional T1 and frame relay interfaces. It's taking shape as internetworking devices (bridges and routers) with built-in wide-area capabilities and traditional wide-area network equipment (muxes, nodal processors, etc.) that have built-in internetworking capabilities.

The emergence of distributed applications and availability of cheaper T-1 and fractional T-1 are helping to push the interconnection of local networks over the wide area. The most important driving force, however, is an increasingly sophisticated corporate management demanding corporate-wide, cost-effective, any-to-any communications.

The importance of LAN interconnection has not been lost on the vendor community. Drew Major, one of the lead Novell NetWare architects, recently noted that improving NetWare performance over the wide area tops his 1991 enhancement priority list. The Open Software Foundation has endorsed Andrew File System over Sun's (Mountain View, CA) Network File System, citing the Andrew File System's ability to scale over the wide area.

In the coming era of WANs linking LANs, T1 will be so cheap that most Fortune 500 and many Fortune 200 companies will be able to afford a private data network.

Planners considering network interconnection strategies face a variety of switching and interface options. Critical issues in LAN-to-WAN interconnection are how data is moved from the bridge (or router) to the backbone, and how data is moved between nodes on the backbone. Planners must choose between the less-than-optional approaches that were

developed for the traditional voice network and the promise of solutions that may take years to materialize.

The frame relay interface, standardized by the American National Standards Institute (ANSI) and International Telegraph and Telephone Consultative Committee (CCITT), possesses the rare combination of immediate availability and being well adapted for LAN-to-backbone interconnection.

Frame relay is garnering wide-spread acclaim as the premier method for interconnecting LANs, but it is crucial to have an understanding of the pros and cons of partnering frame relay with the established internodal switching techniques of circuit and packet switching, as well as StrataCom's FastPacket approach.

A New Wide-Area Architecture

Traditionally, integrated voice and data networks are implemented with rigid T1 nodal processors connected to private-line T1 circuits as can be seen in Figure 1 below.

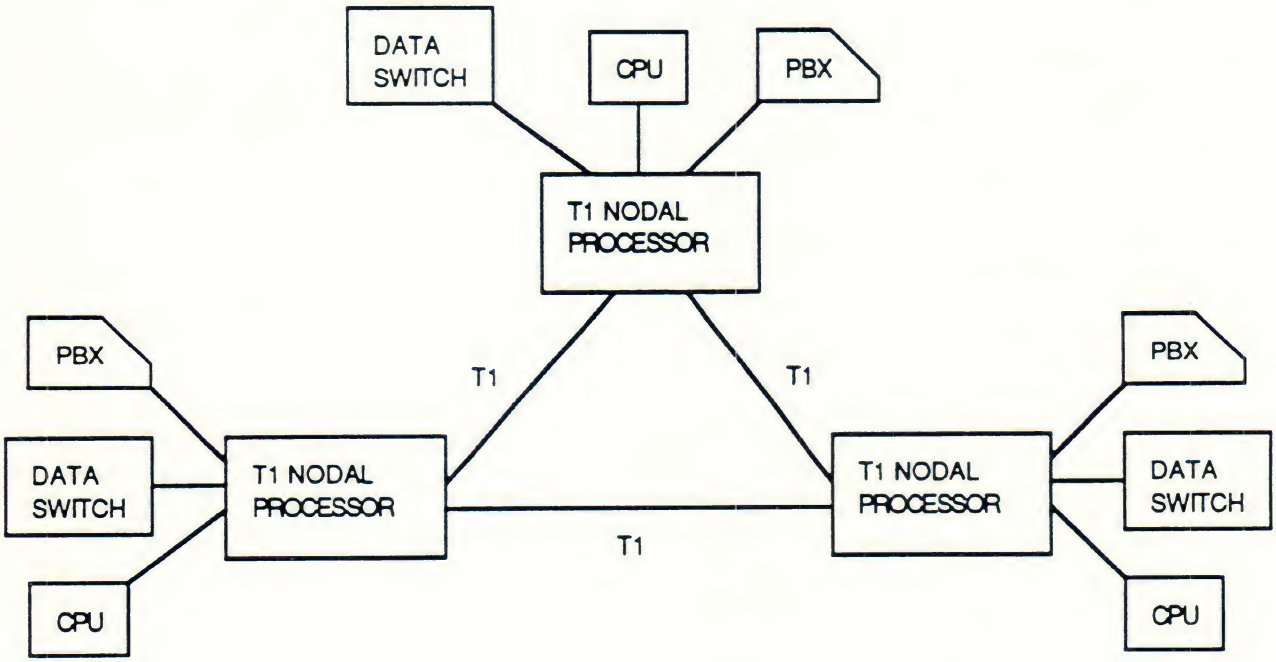


Figure 1 - Integrated voice and data networks of the 1980's

But on today's larger sites, the trend is to split off voice onto a cost-effective IXC virtual private voice network (see Figure 2).

In this scheme, the T1 multiplexor now provides access to public-based voice switching and multiplexed services, while the internetworking traffic is transported over its own private T1 lines.

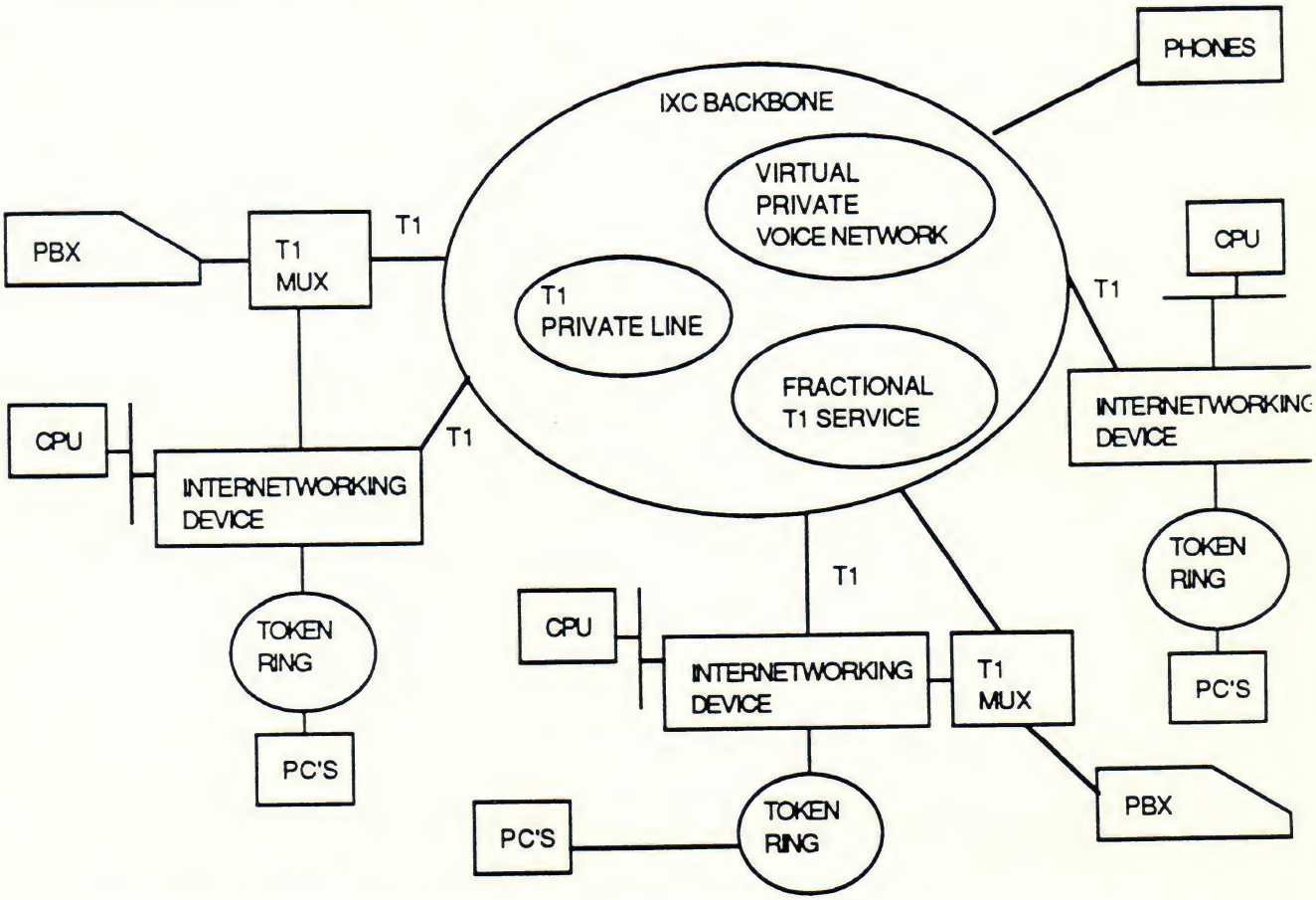


Figure 2 - Emerging wide area networks of the 1990's

But despite the economic and managerial advantages of this approach, it still falls short of the ideal. At the moment, the LAN-to-WAN interconnect is bandwidth-limited by either the private multiplexors or public carriers that link to the internetworking device.

This is particularly a problem with complex LAN interconnect configurations, where many sites need to talk to one another. For instance, creating a mesh network in which each LAN connects to every other still calls for an unacceptably high number of physical interfaces and T1 access links.

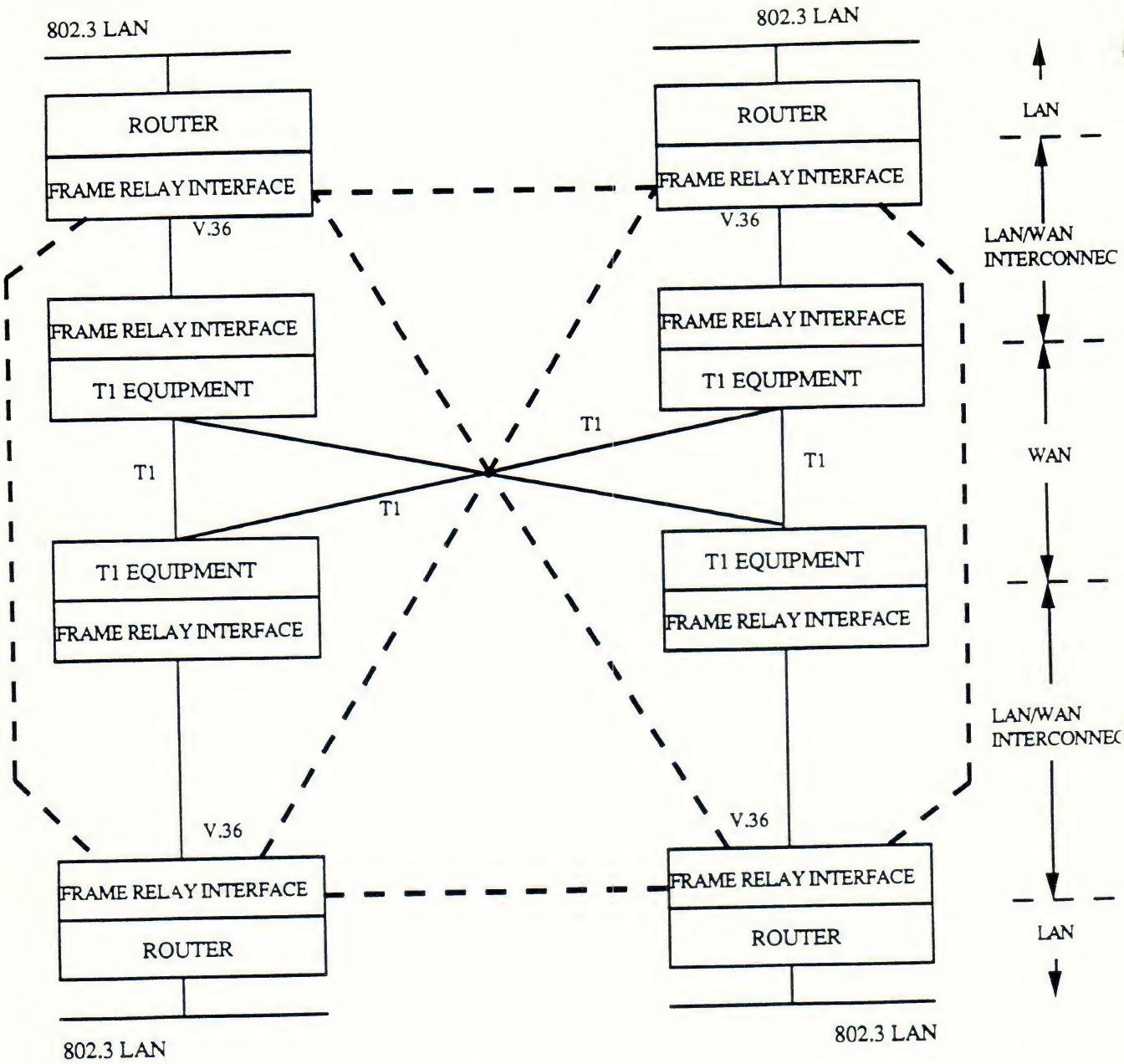


Figure 3 - The Frame Relay Interface

As a wide-area interface and potential public service, frame relay promises to improve wide-area network performance, stream-line interconnects, use bandwidth more efficiently, and reduce overall equipment cost. ***Ultimately, frame relay promises nothing less than a new architecture for the wide area,*** with say, a router, bridge, or hybrid internetworking device furnishing the frame relay interface that serves as the LAN-WAN interconnect as can be seen in Figure 3 above.

What is Frame Relay?

Figure 4 shows a network of four LAN routers interconnected by conventional T1 multiplexors. A separate connection is required between each router. The result is three physical links between the routers and each T1 multiplexor. The problem gets worse as the network grows.

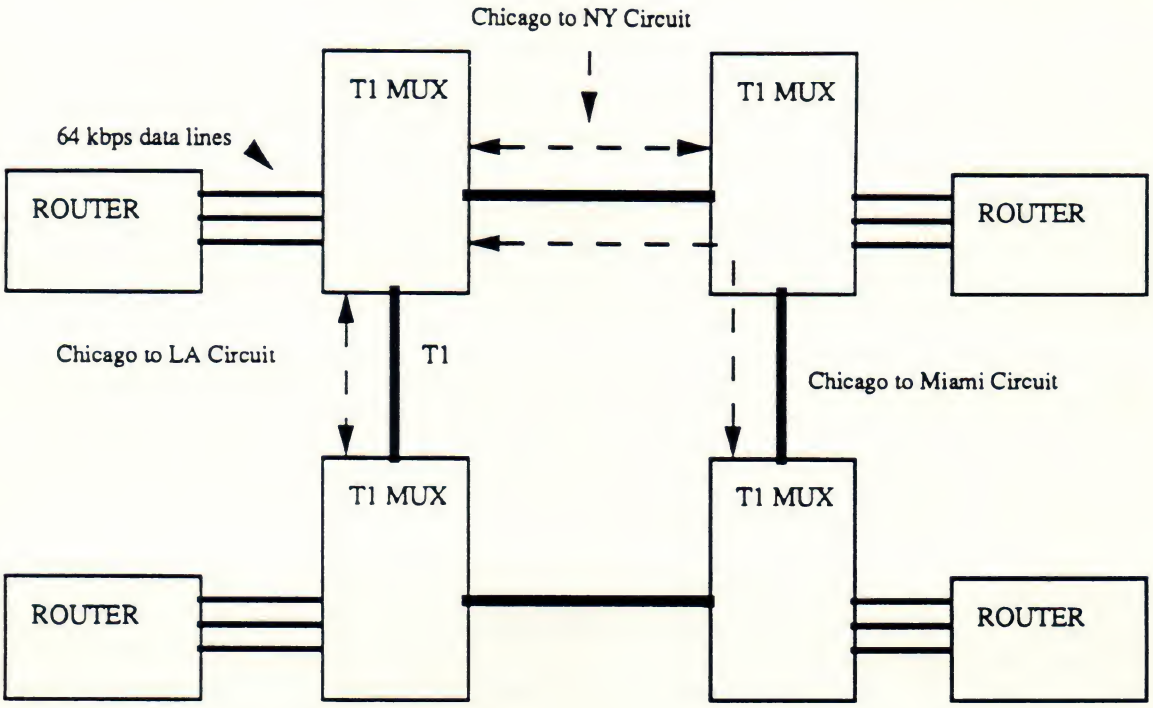


Figure 4 - T1 Backbone using conventional networking T1 multiplexors

In conventional packet switching (ie., X.25), when a data call is established between two devices, a virtual circuit identifier is negotiated and used throughout the call to identify data packets. The routing of these data packets through the network is done at the network layer (Layer 3) of the OSI model.

With LANs and LAN bridges, the routing of the actual data packets is moved down the OSI protocol stack to the data link layer (Layer 2). LAN data packets are stamped with a physical 48-bit destination address; this physical address is used to route the data packet to its destination.

The goal of frame relay is to move the virtual circuit identifier currently implemented in Layer 3, down to Layer 2 so that the wide-area packet switching can be accomplished with the same kind of performance now realized on LANs. The term "relay" is used to imply that as with LAN bridge networks, Layer 2 data frame is not terminated at each switching node but is relayed to its destination. Unlike packet switching, in frame relay the physical line between nodes consists of multiple data links, each with its own flow control windows identified by the address of the data link frame.

Two Flavors of Frame Relay

Frame relay comes in two versions: Frame relay 1 sets up a permanent virtual circuit, in which the end user has no control of connection. Frame relay 2 will offer Switched Virtual Circuit Service (SVCS) but has not yet been implemented.

CCITT Recommendation I.122 - "Framework for Providing Additional Packet Mode Bearer Services" - describes an architectural framework for two types of frame relay services: Frame Relaying 1 and Frame Relaying 2.

Frame Relaying 1 provides the unacknowledged transfer of frames, with routing based on the address carried in the frame. The frames are defined by I.441. Only the framing and address functions of I.441 are terminated, sufficient to interpret the frame address.

Frame Relaying 1 can be provided over switched virtual circuits (SVCs) or permanent virtual circuits (PVCs). The SVC version uses an extended form of I.451 in the ISDN signaling channel for call control. WIth PVCs no real-time call establishment is necessary because the address fields are agreed on when the customer subscribes.

By contrast, in Frame Relaying 2 endpoints always implement the core functions - the complete I.441. Consequently, in Frame Relaying 2 the network may use Layer 2 parameters to facilitate network operations such as charging and resource allocation. In Frame Relaying 1 the network has no knowledge, in principle, of the protocol used end to end.

The main obstacles to quick adoption of frame relay are the lack of products that can act as frame relay nodes and the large installed base of conventional packet switches and routers. But the LAPB frame format can provide a frame relay service that conforms to CCITT I.122 description of Frame Relaying 1 service with PVCs. Although I.122 is oriented toward defining frame relaying over ISDN B-channel, this implementation will provide the same service over a V.35 interface. With the PVC option, no signaling channel is needed for call setup.

Putting the Concept into Practice

Suppose that instead of a separate physical circuit between routers for each data link as illustrated in Figure 4, data links are frame multiplexed on a single high-speed physical circuit. The LAPB (HDLC) data frame supports an eight-bit address, of which only two addresses are typically used - binary 1 and 3. The remaining six bits of the address field can be used as the physical identifier of the destination router for this data frame.

The enhancement of the address field of the HDLC protocol and frame multiplexed link are straight forward and compatible with emerging standards, and should have minimal impact on existing router software. The

field extension bit and Command/Response bit of the address field are unchanged. The six-bit "physical address" mapping can easily be extended to accommodate the 13-bit field used in LAPD. The statistically configured address table can be dynamically assigned when ISDN frame relaying protocols become a reality.

Most routers (and bridges) are implemented with a processor bus architecture in which the processor elements are separate from the I/O elements. Data packets are transferred from some global memory area to the appropriate interface card, which adds the data link framing and transmits the packet onto the physical link. These cards can be replaced with a single card that manages the multiple data links between each router and multiplexes them onto a higher-speed physical interface.

The Implications of Frame Relay

1. Each router can be connected via a virtual circuit to every other router, as opposed to also providing a transit function. Thus routers need only enough processing bandwidth to forward and terminate their own traffic - not their own plus network traffic. This reduces the cost of routers.

2. The bursty nature of the direct router-to-router connection, the statistical sharing inherent in the frame relay technique and the fact that data packets use network resources only once mean that significantly fewer backbone trunks are needed for large WANS than is possible with conventional technologies.

3. Network delay is minimized, alleviating restrictions on using terminal-to-host protocols and increasing throughput.

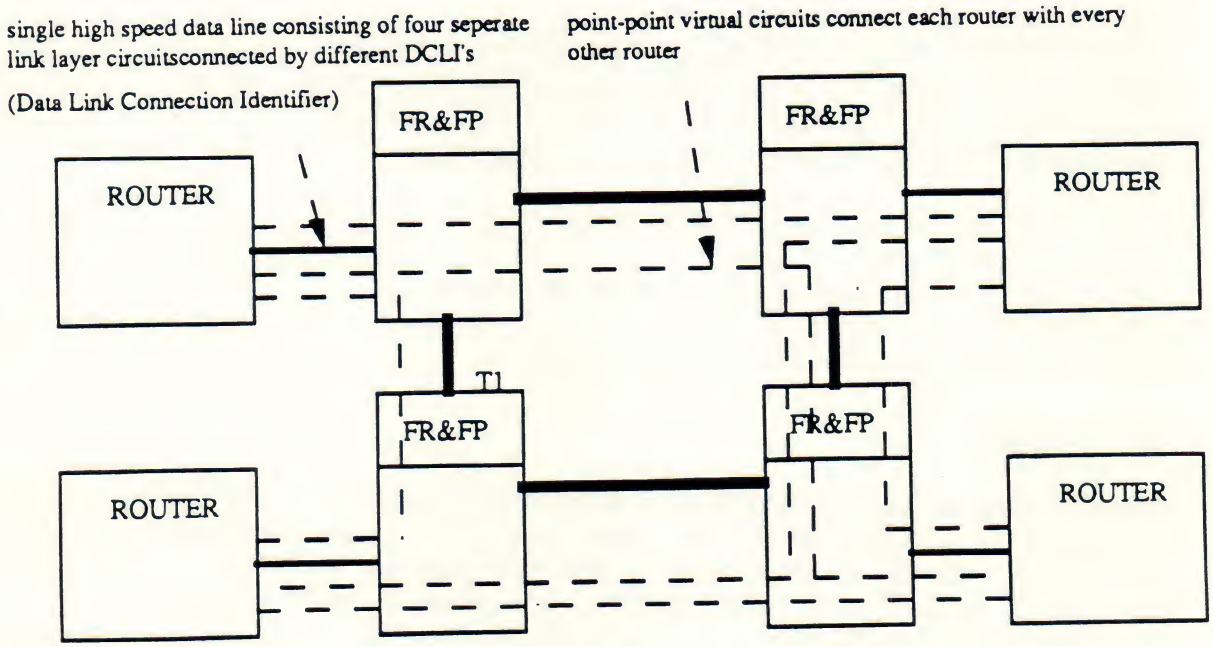


Figure 5 - Fast Packet Switches with Frame Relay

Figure 5 above, illustrates a network with routers, each connected to fast packet switches via a single frame relay interface. Each frame sent from a router contains an address that is used by the switch to correctly route the frame to its destination. The switches perform transit switching, and the frames are statistically multiplexed on the T1 links. This method allows data to be transmitted over the backbone network at T1 speeds rather than at sub-rate speeds, as they would be with conventional T1 mulplexors.

The limitations of circuit and packet switching in the age of image and cooperative processing have been well-documented as have the merits of integrated circuit packet switching. Several forces have limited the success of this new technology, such as rapidly falling bandwidth prices and massive embedded bases of existing technologies. It is believed, however, that heretofore there have been two core gating factors. The first used to be the lack of product.

The second major inhibitor to user acceptance of integrated circuit/packet switching has been the lack of standards for a while. Although the CCITT has addressed this issue with its work on Frame Relay (1988 Recommendation I.122), and solid standards are only now emerging, the real problem for users was that proprietary implementations like "fast packet" sacrificed compatibility for performance. First, these products were not compatible with existing circuit-switched, public-network service offerings like AT & T's M44, M24, etc. Second, the technology performed well for data transport, but there were tradeoffs. For an SNA environment, the technology did not then support digital bridging so a user had to abandon a multi-drop topology or use tail circuits (and thus lose visibility).

It is believed that in the existing and emerging corporate environment these were the proper design decisions to make. We see two trends that this architecture should be able to leverage. First, many corporate backbones are increasingly being employed primarily to carry business-critical data applications, with voice as a secondary consideration. Second, corporations are realizing the limitations of SDLC and the benefits of HDLC, and are looking for non-disruptive migration strategies.

Any corporate communications manager charged with developing or planning for a corporate information network can no longer afford to ignore the concept of integrated circuit/packet switching. Netrix has developed a unique, and we believe compelling, approach to the problem that bears closer examination by enterprises today.

Deliverables

Frame relay/fast packet (FR/FP) is multiplexing technology that permits and supports the deployment of interlocation networks with valuable new attributes. In practical terms, IBM, DEC and other protocols can be mixed with each other, and with voice and video, on the same links without protocol processing or other performance-impairing compromises. In addition, because of the statistical interleaving of the load, dramatic

improvements in throughput performance over traditional multiplexing techniques are attained. Perhaps more importantly, since FR/FP will soon be a recognized standard, AT&T already offers a precursor service by means of which multiplexing, and access to conventional voice and data as a tariffed service. This greatly expands the geographic scope of dispersed networks, and reduces their manpower, space and capital-equipment requirements. Thus, the first key deliverable: higher function, lower-cost networks.

How it Works

In the simplest terms, FR/FP accepts packets or frames offered to it and packetizes traffic not already packetized, adapting its packet size to fit the offered frame (so as to avoid fractured SDLC frames). FR/FP then adds three preambles: an address containing an explicit route, a content preamble and a handling-priority preamble. The content and explicit route address allow the transmission of traffic without store and forward, and readdressing delay, thus supporting voice and other forms of synchronous traffic. The handling preamble coupled with the content preamble permit, on a user-controlled basis, dynamic reduction in the effective bit rate of the voice traffic and deliberate delays of up to one or more seconds in the data traffic to increase the effective busy-hour throughput of the circuit. It is here that genuine improvements, on the order of 5-to-1 for T1-sized circuits, in peak-hour throughput are achievable. This has major ramifications in the local loop. That area, still largely under the control of monopoly local carriers, has evaded productivity increases, and it is there that FR/FP has its greatest user bandwidth multiplication (cost reduction) impact.

WAN Bandwidth on Demand

Frame relay is an ISDN spin-off that eliminates much of the time-consuming network processing associated with today's X.25 packet networks. Frame relay is extremely fast because it uses a bare minimum of routing functions in what is essentially a Layer 2 protocol derived from CCITT's link access protocol for the ISDN D-channel (LAPD). Unlike X.25, frame relay concentrates its error checking at the end nodes, freeing intermediate nodes to forward packets faster, in much the same way that the data link layer works on a LAN. Also unlike X.25, frame-relay services will support both voice and data.

So efficient is this new networking scheme that many of its users boast of speed 10 times that of standard packet switched networks using the same hardware. For network managers, frame relay promises to open up a new world of high-speed voice and data services that do not require commitment to physical T1 private network.

X.25 Networks Losing Momentum

Today's packet networks employ a large number of low-speed connections, switched on a backbone of high-speed links. But with the advent of fibre optic technology and improved digital-signaling techniques, general packet-

switched networks are looking to the future with new technologies in mind.

Traditional wide area networks have been designed to handle voice or data, but not both. The new specially designed frame-relay engines, however, have been streamlined to handle voice equally as well as data.

Another reason for the lack of details on frame relay is that much of the technology, even though based on standards, is still proprietary. Hence most of the early frame-relay vendors are maneuvering within the CCITT Study Group XVII to be the recognized standard-bearers, or failing that, to have a good portion of their network features standardized.

Large network users believe that network management can get out of control with virtual circuit technology. Owing to the nature of virtual circuits, it is nearly impossible to run network diagnostic gear on a network link, since any given call is liable to be disconnected by the time a consistent error is experienced. This leaves major responsibility for end-to-end network integrity with the carriers.

Telco Packets: A good example of this is AT&Ts integrated Access and Cross-Connect System (IACS), based on the IACS frame-relay switch. (AT&T prefers the term wide-band packet switching as opposed to frame relay). AT&T is planning to sell the switches to the local exchange carriers (LECs), of which only Nynex is talking about providing a frame-relay service. AT&T has not said whether it will provide interLATA frame relay service to connect local offerings.

The hope is to get the LECs back into the private- network market (with IACS).

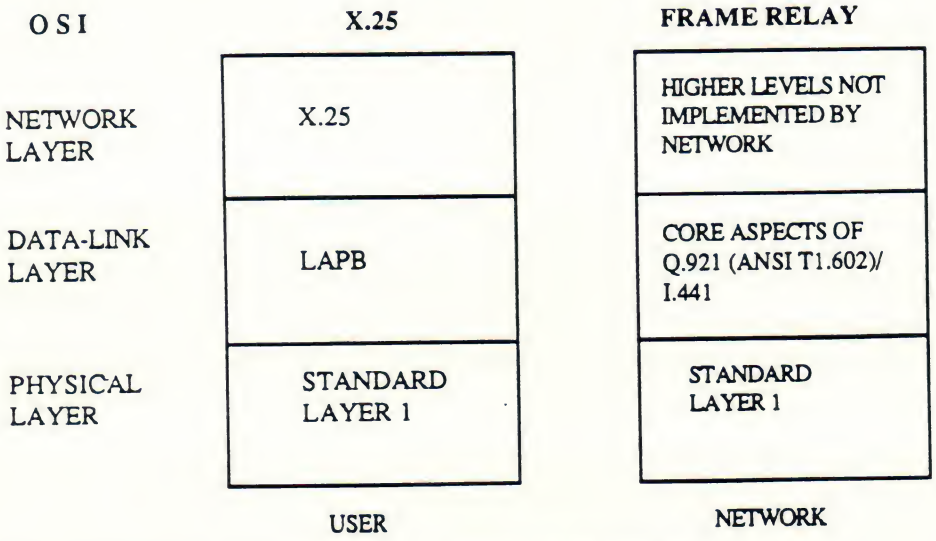
One of the unique features of the upcoming IACS is that it can sense the difference between voice and data. Voice is digitized using an adaptive differential pulse-code modulation (ADPCM) like packet voice network protocol currently under study by the CCITT Study Group XVII. The IACS achieves about a 4:1 voice compression ratio.

With data the IACS can vary its speed and can assign variable-length packets to each data stream, depending on the bit rate. For example, a 9.6-kbit/s data call could be assigned a 20-kbit/s packet rather than a 64-kbit/s packet, thus causing less overhead, and allowing more data on the network as needs demand. Currently, the planned access to AT&T IACS is via a DSI (1.54-Mbit/s) superframe or extended superframe format.

Lean is Better

Frame relay is a standards-based interconnecting method created primarily to link LANs to the WAN backbone. Its benefits include low overhead, high capacity (up to 2Mbps) with low delay, and reliable data transfer over today's (digital, intelligent, and reliable) networks.

Frame relay achieves its high throughput and low delay by eliminating the overhead of error correction that unreliable voice circuits of the past required. Frame relay only engages in error detection at the first two layers of the OSI model X.25, on the other hand, uses a full three-layer OSI interface (See Figure 6). X.25, designed to work in dirty transmission environments, uses up to two-thirds of its state tables for error policing functions.



Frame relay engages in error detection only at the first two layers of the OSI model.
Frame relay takes advantage of today's digital, intelligent, and reliable networks.
X.25 uses a full 3-layer OSI interface, which causes a higher overload.

Figure 6 - X.25 and Frame Relay Stacks

Given that frame relay achieves high throughput by divesting itself of error detection and other functions, you may ask how error detection is performed in the frame-relay environment. This illuminates two assumptions about frame-relay implementation: First, in the frame relay environment, error detection and retransmission functions are pushed out to the user equipment. Terminal equipment must be sufficiently intelligent to detect the receipt of errors and to request end-to-end retransmission. Second, frame relay is essentially designed for high-quality, reduced-error transmission environments. It is not suited for error-prone analog facilities.

These factors weigh heavily in any frame-relay decision where analog transmission facilities comprise significant portions of the network, and where a large installed base of dumb end-user equipment exists.

The Origins of Frame Relay

Frame relay was not initially developed as a solution to LAN-WAN interconnection; it was a byproduct of work done on the Integrated Services Digital Network (ISDN) standard. In the development of ISDN, a packet

bearer mode was defined to provide multiplexing individually addressed units at layer 2 rather than allocating time slots. Link Access Protocol for ISDN D channels (LAPD) is defined by the CCITT Q.921 standard. It was felt that multiplexing at layer 2 would be viable technology beyond its use in ISDN, and the CCITT I.122 committee was formed. The corresponding ANSI committee formed T-1.606 to handle the Frame Relaying Bearer Service.

Additional specifications dealing with addressing and congestion control have been proposed by ANSI and forwarded to CCITT. The Local Management Interface transfers information between the user device (router) and the network.

Addressing in Frame Relay

Addressing in frame relay is handled by an 11-bit address field called the Data Link Connection Identifier (DLCI). By offering independent packet addressing, frame relay reduces the fixed circuit requirements, which made traditional hierarchical networks overhead-intensive. It allows private virtual circuits to be set up between LANs without adding delay between the nodes.

The DLCI supplies the virtual circuit number corresponding to a particular port on a bridge or router to which the receiving LAN is attached. A sending device simply places an addressed frame onto the network where it is directed to the receiving device. In this sense, frame relay WANs behave more like today's LANs than the long haul networks of the past.

Congestion Control

With the bursty nature of LAN-to-LAN interconnection, it's quite possible to overload the receiving devices' buffers. As a result, ANSI addressed the congestion control issue by providing options to the frame relay specifications. Explicit Congestion Notification (ECN) offers the ability to communicate traffic overload problems to downstream nodes (forward ECN) and upstream nodes (backward ECN).

Forward and backward ECN do not solve the entire problem, however. Because of frame relay's philosophy of low overhead and error detection - not correction - the basic frame relay specification does not allow packets to be sent upstream. The receiving node does not have a vehicle for informing the sending node that it is experiencing congestion.

Consequently, ANSI defined Consolidated Link Layer Management (CLLM), which reserves the address DLCI 1023 for the network to send congestion alerts to user devices. Network managers can also provide for Implicit Congestion Control (ICC), which is defined by upper layer network protocols, such as TCP-IP.

ANSI is currently working on specifications to provide the private virtual

circuit's status. This will reveal whether the circuit is up, down, or congested as well as the status of valid DLCIs assigned to that private virtual circuit.

The Circuit Switching Option

Frame relay specifies how data is placed onto the WAN backbone. The network may use a transmission method to move data between nodes. The transmission method can either maximize frame relay's benefits or degrade them. Circuit switching, packet switching, or FastPacket switching may be used (see Table 1) as the transmission method.

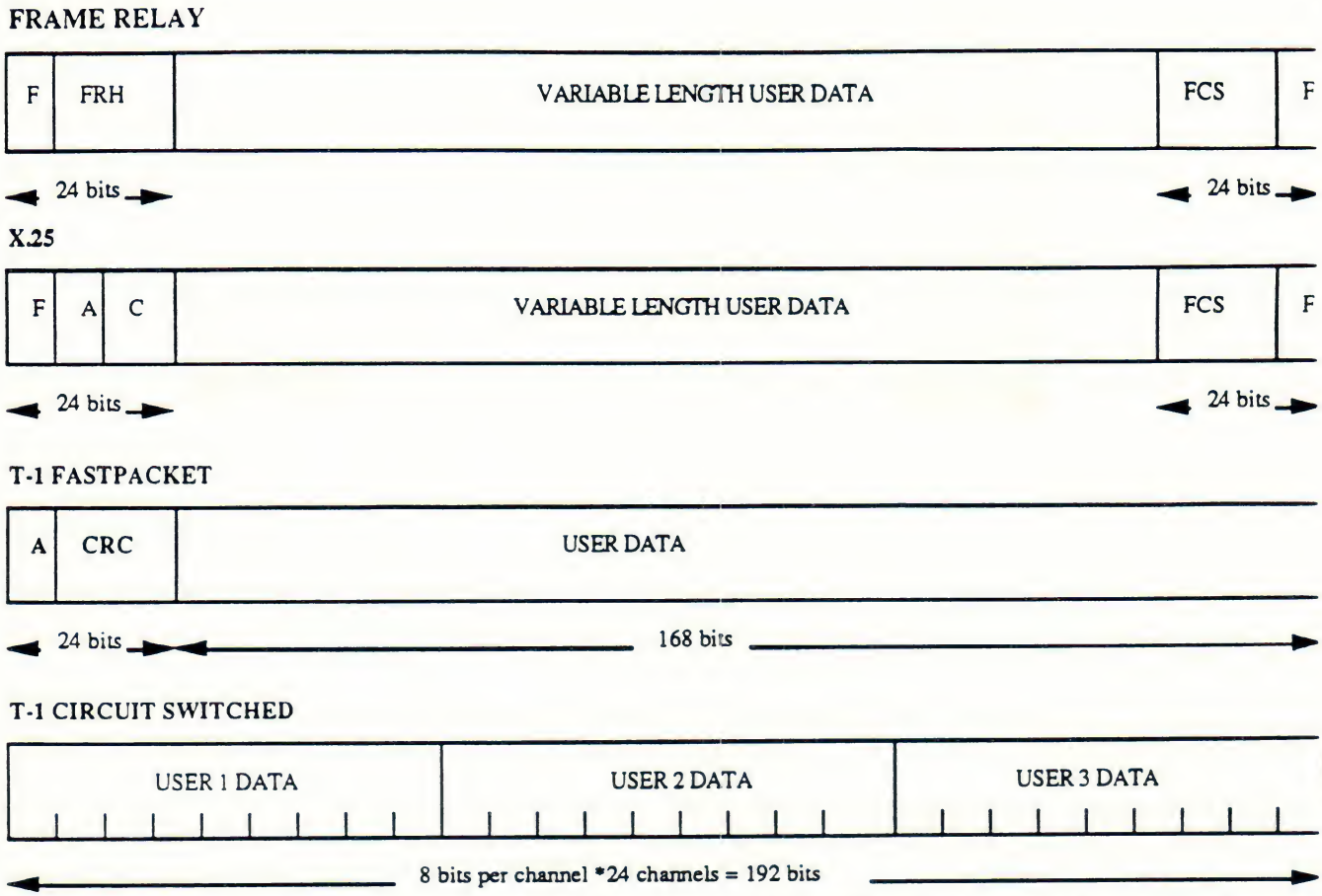
	CIRCUIT	PACKET	FASTPACKET
Throughput	HIGH	LOW	HIGH
Delay	LOW	HIGH	LOW
Channel	CLEAR	PACKET	CLEAR
Routing	SESSION	PACKET	PACKET
Bandwidth Allocation	FIXED	DYNAMIC	DYNAMIC
Signalling Channel	FIXED	DYNAMIC	DYNAMIC

Table 1 Switching Techniques Compared

Circuit switching is a means of sending multiple data channels on a single physical line. This is accomplished by Time Division Multiplexing (TDM), or assigning each channel a specific portion of the available bandwidth (see Figure 7). In circuit-switching, each channel's bandwidth is pre-assigned and dedicated to that application. When frame relay is tied to a circuit-switched device, the frame relay portion is assigned a channel in the TDM frame, just as any other data connection.

Channelling works well for steady streams of frame relay data but it makes it difficult to take advantage of one of frame relay's primary advantages: the efficient handling of bursty data. To accommodate the data bursts, the circuit switch channel must be large enough to handle the maximum burst capacity of the frame relay device. Even with declining costs of bandwidth, this is an expensive solution. During normal operation, the excess

bandwidth is not being used by the frame-relay channel and is also unavailable to the other circuits because the data is sent in specific channels.



Frame relay specifies how data is placed onto the WAN backbone, not the underlying transmission method. It can use circuit switching or FastPacket to move data between nodes. each can either maximize frame relay's benefits or degrade them.

F=Flag FCS= Frame Check Sequence FRH=Frame Relay Header A=Address C= Control

Figure 7 - Data Packet Types Compared

The Packet Switching Option

An alternative to sending data in specific channels is to segment the data and send it across the network in packets. One method of packet transmission is X.25 packet switching, which allows reliable transmission over the public X.25 network, typically at speeds up to 64Kbps. X.25 is a full-featured protocol that operates at OSI layer 3 and encompasses data encapsulation and transmission, error correction and retransmission, network sychronization, and flow control (see Figure 7).

In an X.25 network, all bandwidth is available to any channel that is generating traffic. If multiple channels request the bandwidth at the same time, the X.25 switch chooses which packet to send first and buffers the

other packets until bandwidth becomes available. This is an excellent data transmission method when the physical transmission layer is unreliable or noisy because the error correction is built into the protocol. As the problem of noisy and unreliable lines has almost disappeared on today's digital networks and as computers have begun handling their own error detection and retransmission, the associated overhead and delay incurred performing these functions is becoming unnecessary in most cases.

The working promise of frame relay is that network transmission will take place in the near-flawless environment of most digital backbone networks (one error in every 10^6). With this kind of performance, OSI Level 2 packets may be passed through the network with a minimal number of errors.

If errors occur at Level 2 they are simply discarded, with the assumption that higher-layer protocols at the end nodes of the network will recover them.

The need for high speed bridging of lans, and the ability to pump data in various ways down pipes of two Mbit/s capacity or more, has led to a number of developments on the X.25 front over the last couple of years. The only trouble is that, in some cases, this has lead to a non-standard approach.

This is certainly true of fast packet switching, one of the latest developments introduced to overcome X.25's inability to provide high speed links. Fast packet switching combines the advantages of X.25 (efficient utilization of 64 kbit/s channels) with the speed of time division multiplexing and circuit switching over trunks.

The advantage of X.25 is you're effectively statmuxing a number of calls onto a channel. TDM has the advantage of high speed, but may not be making high utilization of the bandwidth. Fast packet switching can also carry voice over trunk backbone circuits. Effectively you can have voice on X.25, using the same type of statistical multiplexing over high speed links.

The reason it can do this is that fast packet switching is not really X.25 at all; it supports X.25 speeds by ignoring some of its addressing aspects. Currently, in fact, there are no CCITT standards for fast packet switching, and at least two different, incompatible approaches have appeared.

The first approach involves a frame relay process whereby packets are quickly forwarded over trunk circuits by ignoring X.25 headers. The other approach uses a different technology to pump packets of data along a high speed pipe, utilizing the whole bandwidth if necessary. This could be used to link lans which also use data packets (such as Ethernet) across wide area links, to provide much higher throughput than offered by current lan routing technology.

THE FASTPACKET CHOICE

Another way to transmit packets is the "FastPacket" method. FastPacket

strips most of the error checking and all of the retransmission characteristics out of X.25 and provides a shortened header to achieve the lowest overhead possible for a packet network (see Figure 7). This method only provides a data-integrity check in the form of a cyclic redundancy check, and it will discard packets it finds to be corrupted. It relies on the end nodes to request retransmission of the lost packets and to handle most of the flow control. The advantage of offloading these functions is that the network can operate at very high speeds (tp to 2Mbps) and still provide bandwidth on demand to handle the bursty characteristics of LANs.

FastPacket, like frame relay, was developed to exploit the greater intelligence and reliability of today's networks. FastPacket is touted as the best current switch method for the frame relay interface (see Figure 3). The closer the match between the internodal transmission and the LAN interconnect, the better the native efficiencies of frame relay can be exploited.

FastPacket is neither a product nor an interface specification. Rather, it is a technique for asynchronously transferring data across the link. FastPacket provides fixed packets that are given access to the entire T-1 bandwidth rather that 24 channels as in traditional TDM switches. FastPacket is a registered trademark of StrataCom. StrataCom makes the IPX multiplexor, which is also resold by Codex (Mansfield, Ma), Digital Equipment Corp, (DEC, Maynard, Ma), and others. It defines a fixed length packet of 192 bits (the final bit is for synchronization).

The proprietary term "FastPacket" should not be confused with the generic term "fast packet". The latter term refers to various streamlined packet technologies including frame relay, Broadband ISDN, cell relay, and Asynchronous Transfer Mode. Each of the "fast packet" methods combines the low delay and high throughput of circuit switching with the packet-level dynamic routing used in packet switching.

Hybrids

The advent of fast packet switching heralds from the next era of voice/data integration over hybrid trunk networks. In addition, equipment suppliers are introducing ISDN support on X.25 switches and conversely, pabx suppliers such as GPT are introducing packet handlers. This means users can have a single network with just one set of devices for voice and data switching at each site, as well as common trunks between them. The question is whether to opt for the pabx/packet handler solution or to use X.25 switches that support ISDN to carry voice. The problem with the packet handling approach is that it uses up the resources of the pabx; most users may prefer to keep separate pabxs, and link voice an data networks through ISDN interfaces on the packet switch.

Statistical Multiplexors

Packet switching is not only being combined with voice, but also with other

forms of data networking, notably its close sibling, statistical multiplexing at the local end, and circuit switching over wide area links. As a result, users can mix and match to obtain the efficiency of stat muxing, along with the flexibility and resilience of X.25, while exploiting the full performance of high speed trunks.

At the local end, its common to transmit data from several terminals within a single packet using statistical multiplexing, to make more efficient use of bandwidth.

In theory, up to 512 users can be multiplexed down a single channel at data rates from 2400 bits/s to 72 Kbits/s, although in practice a network would probably not operate efficiently with such a large number sharing a single link.

CHAPTER 2 - FRAME RELAY SERVICES DESCRIPTION

Introduction

The separation of the user and control planes for all telecommunication services has been established as a fundamental principle of ISDN protocol reference model (CCITT Recommendation I.320 Red Book). However, this principle has been applied only to circuit-mode services. Packet-mode services in ISDN are based on CCITT Recommendation X.31. Recommendation X.31 is a pragmatic approach that minimizes deployment and interworking difficulties, while at the same time providing access to packet services through an integrated physical interface.

The evolution of packet-mode services in ISDN has been investigated, and an architectural framework for providing additional packet-mode services has been established. In undertaking this investigation, the objectives were to define a framework that is based more fully on the ISDN protocol reference model there by achieving:

1. Full integration of C-plane (control plane) procedures for all services, that is, one set of protocols for call control, supplementary services, operational, administration, maintenance, and provisioning functions across all telecommunications services.

2. The decoupling of user information transfer requirements from C-plane transfer requirements. This allows the possibility of defining telecommunication services whose U-plane characteristics are tailor-made only to the user information transfer needs, and not to requirements needed for the transfer of C-plane information.

3. Improved transit delay and support of a higher throughput when compared with packet-mode services based on CCITT Recommendation X.31.

The bearer services defined in this standard are of the (frame-relaying) virtual call and (frame-relaying) permanent virtual-circuit type.

FRAME-RELAYING SERVICE DESCRIPTION

Quality of Service Parameters -

The quality that frame-relaying service provides is characterized by the values of the following parameters (the specific list of negotiable parameters can be taken up as an extension to this study):

1. Throughput
2. Transit Delay
3. Information Integrity
4. Residual Error Rate
5. Delivered Error Frames
6. Delivered Duplicated Frames
7. Delivered Out-of-Sequence Frames

8. Lost Frames
9. Misdelivered Frames
10. Switched Virtual Call Established Delay
11. Switched Virtual Call Clearing Delay
12. Switched Virtual Call Establishment Failure
13. Premature Disconnect
14. Switched Virtual Call Clearing Failure
15. Others for Further Study

NOTE: The Values of these parameters are dependant on particular implementations. This standard does not specify their values. These parameters are subject for future standardization.

Service Characteristics

The basic bearer service provided by the network is the order-preserving transfer (see below) of frames from the network side of one user-network interface. The frames are routed through the network on the basis of an attached label (eg., the Data Link Connection Identifier (DLCI) value of the frame). This label is a logical identifier with local significance. In the virtual call case, the value of the logical identifier and the other associated parameters (eg., layer-1 channel, delay, and so forth) may be requested and negotiated during the call set-up by means of C-plane procedures. Depending on the value of the procedures, the network may accept or reject the call. In the permanent virtual circuit case, the logical identifier and the other associated parameters are defined by means of administrative procedures.

The user-network interface structure allows for the establishment of multiple virtual calls or permanent virtual circuits, or both, to many destinations over a single access channel.

More specifically, for each connection in the U-plane the bearer service:

1. Provides bidirectional transfers of frames.
2. Preserves their order as given at one user-network interface if and when they are delivered at the other end.

   NOTE: Since the network only terminates the core functions of CCITT Recommendation Q.921 (ANSI T1.602-1989), no sequence numbers are kept by the network. Networks should be implemented in such a way that frame order is preserved.

3. Detects transmission, format, and operational errors (for example, frames of unknown label).
4. Transports the user data contents of a frame transparently; only the frame's address and FCS fields may be modified.
5. Does not acknowledge frames.

All of the above functions are based on CCITT Recommendation Q.921 (ANSI

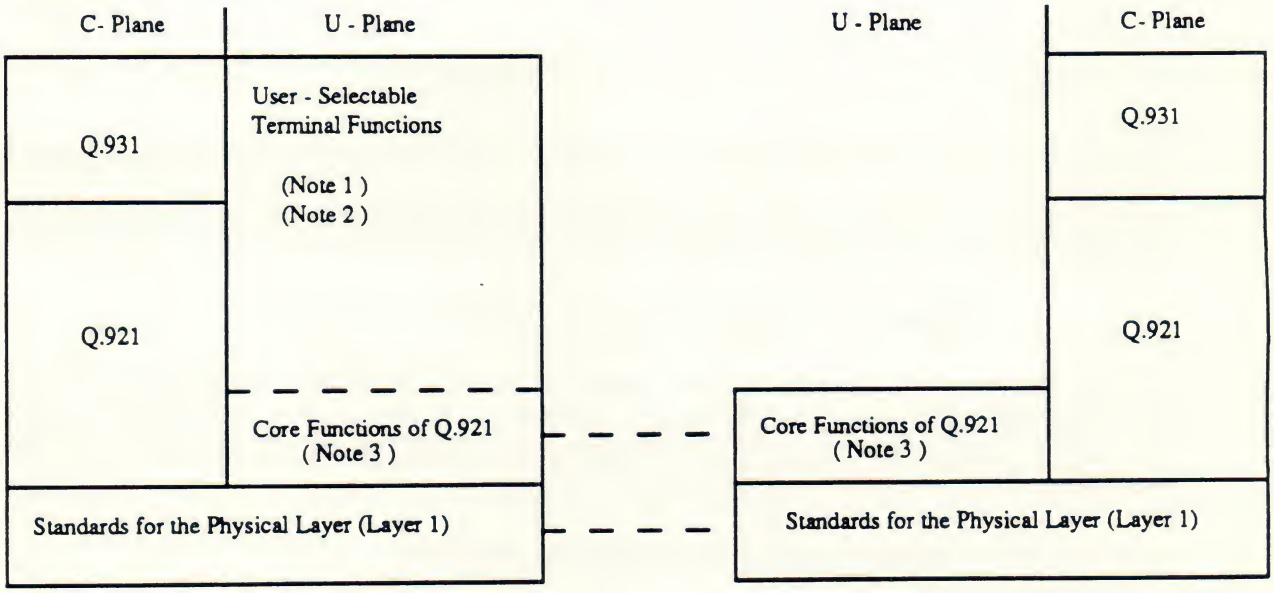
T1.602-1989). Appropriate extensions to the core functions of Q.921 (ANSI T1.602-1989) may be needed, for eg, for congestion control. Additional functions may be needed for throughput monitoring and enforcement. The mechanisms to achieve these are still under investigation.

In the C-plane, all signaling capabilities for call control, parameter negotiation, and the like, as an objective should be based on a set of protocols common to all ISDN telecommunication services. In the case of permanent virtual circuits, no real-time call establishment is necessary and parameters are agreed upon at subscription time.

Appropriate protocol capabilities should be available so that the network may discard erroneous frames if it elects to do so. Note that networks should attempt to discard erroneous frames in order to minimize the possibility of fraud and misdelivery of frames.

USER-Network Interface Protocol Reference Model

Figure 1 is a direct application of the ISDN protocol reference model to the frame-relaying bearer service defined in this standard. It shows the user-network interface protocol architecture. The functions shown on the network side can be accessed from the user side of the user-network interface.



NOTES:
(1) - These functions are user-selectable
(2) - Additional requirements may be placed on terminals depending on the congestion control and throughput enforcement used. One mechanism that can satisfy the congestion control requirements is for the terminal to implement extensions to CCITT Rec. Q.921 to include the dynamic window algorithm.
(3) - The core functions of CCITT Rec Q.921 are defined in 6.3.1

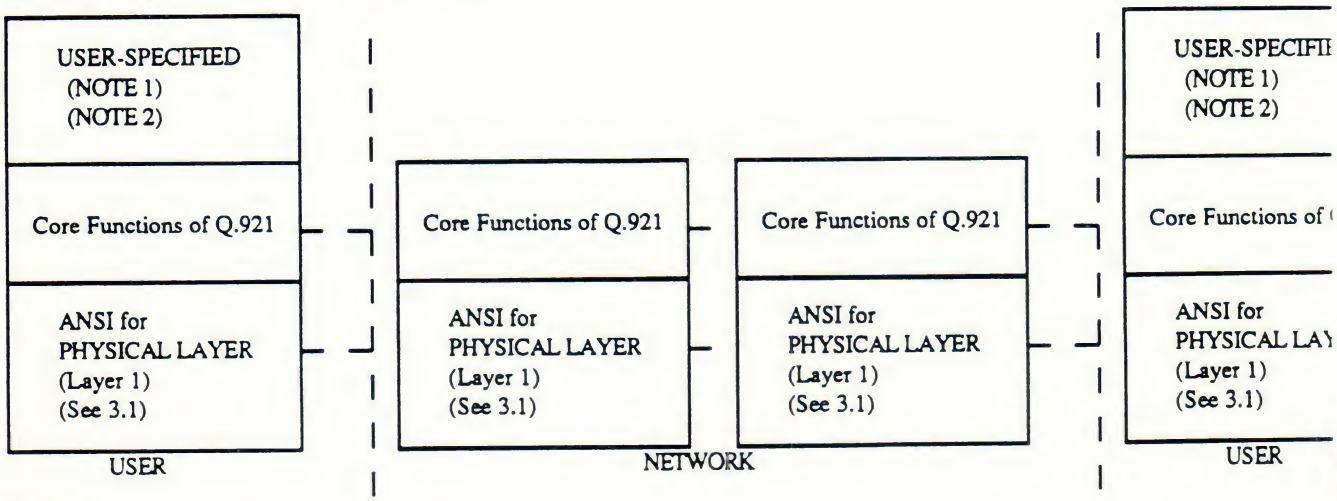
Figure 1 - User-Network Interface Protocol Architecture

The frame-relaying bearer service is provided when no user functions above

the core functions of CCITT Recommendation Q.921 (ANSI T1.602-1989) are terminated in the network; if needed, such functions are terminated only end-to-end.

NOTE: Additional requirements may be placed on terminals depending on the congestion control and throughput enforcement used. See Fig 1.

On the user side, the American National Standards for the physical layer (layer 1) provide the layer-1 protocol for the U-(user) and C-(control) planes. The C-plane uses the D-channel with CCITT REcommendations Q.921 (ANSI T1.602-1989) and



NOTES:
(1) One example is extensions to CCITT Rec Q.921 to include dynamic window algorithm. Other standards or proprietary protocols ma also used.
(2) Additional requirements may be placed on terminalsdepending on the congestion control and throughput enforcement used.
(3) Additional functions may be needed for throughputmonitoring and enforcement. The mechanisms to achieve these are still under stu

Figure 2 - Frame Relaying Service : U-Plane

In the following sections, the designators ISDN (FR) and ISDN (X.31), and the like, do not necessarily imply different ISDNs. For descriptive purposes, an ISDN having the needed capabilities is identified, on a per call basis, as an ISDN (FR), or ISDN (X.31) and the like.

Interworking Between Frame-Relaying Service

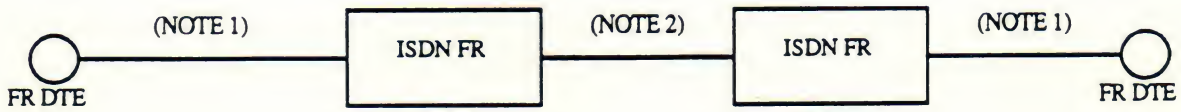
Internetwork Interworking Arrangement - Figure 3 shows the internetwork interworking arrangement for interworking configuration(1). This arrangement applies either when both networks are in the same country, or when they are in different countries. At the internetwork reference point:

1. The U-plane procedure can be based on the core functions of CCITT Recommendations Q.921 (ANSI T1.602-1989). Appropriate extensions may be required, eg., for congestion control.

2. For the C-plane, there are two cases:

a) In the virtual call case, either SS7, or the combination of Q.921 (ANSI T1.602-1989) (layer 2) and Q.931 (layer 3) can be used. Appropriate extensions are required in either set of procedures; extensions to SS7 to cover frame-relaying requirements, and extensions to Q.921 (ANSI T1.602-1989) and Q.931 to cover internetwork interface requirements (Eg., transit network identification). The choice between SS7 or Q.921 (ANSI T1.602-1989) and Q.931 is for further study.

b) In the permanent virtual circuit case, the C-plane procedures are defined by means of administrative arrangements. Signaling requirements are for further study.



NOTES :
(1) U-Plane: Core functions of Q.921
C-Plane: For virtual calls Q.921 and Q.931
(2) U-Plane: Core functions of Q.921 with appropriate extensions
C-Plane: For virtual calls
Option 1: SS7
Option 2: Q.921 and Q.931 appropriate extensions are required for both options

Figure 3 - Internetworking between Frame-Relaying Service

Internal Interworking within a Network

It is advantageous that the U-plane and the C-plane procedures at an internetwork reference point be developed in such a way that they also could be used within a network that offers frame-relaying service. The use of these procedures within a network is entirely at the discretion of the network provider; it is not intended to specify interfaces with associated procedures.

To aid the development of such internetwork procedures, in the following sections a possible model for the functional structure of a network node is outlined. Such a model is presented here for illustration purposes. The exact details may vary.

Classification of Nodal Functions for Interworking

Nodal functions include the following three general functions:

1. Frame-handling functions
2. Network management functions
3. Testing and maintenance functions

Frame-Handling Functions

A frame of variable length, contains:

1. A label for routing purposes
2. User information
3. FCS check field for error detection
4. Other control fields (to be Defined)

The frame-handling functions of each node jointly ensure that frames accepted from a user source are delivered to the specified user sink. Based on an analysis of the label (and possibly other control fields) of a frame, each frame is directed to the proper link for transmission. For internetworking purposes, frame transfer can be:

1. From an access link to an outgoing link in the case of an entry node
2. From an incoming link to an outgoing link in the case of a tandem node
3. From an incoming link to an access line in the case of an exit node

For network control purposes, there can also be frame transfer from an incoming link to an internal function within a given node.

Network Management Functions

These functions ensure that the performance requirements of frame-relaying service can be met, despite failures of network components and fluctuations in the loading of the network. This is achieved through monitoring the availability of the different links and routes, and control of traffic in case of congestion.

Network management functions can be further sub-divided into four categories:

1. *Link management* - This function is responsible for monitoring the up/down status and error performance of an individual link. After link failure has occurred, it also initiates recovery actions for the restoration of the failed link for frame transmission.

The link management function supplies information to the route management function about the availability of an individual link.

2. *Route management* - Based on the link availability information, and possibly information from other internal nodal functions (Eg, measurement of resource utilization), the route management function of a node makes decisions on the set of outgoing link(s) that can be used for sending frames to a particular destination node.

Depending on the routing strategy to be used, appropriate decisions about route availability may be reached:

1. Individually by each node (isolated routing)
2. Jointly among the nodes in the network (distributed routing)
3. Through consultation with a routing agent (centralized routing)

The route management functions supplies information to the call management function about the availability of different routes.

3. *Call management* - Using the route availability information, the call management function of a node routes a frame-relaying call from the node to a succeeding node during the call establishment phase.

The call management function may or may not bind a call to a route. WHen a call is bound to a route, the call management function of each node assigns to the DLCI (Data link connection identifier) with local significance. SUch a DLCI shall form the label of each succeeding frame from the call. Additionally, appropriate resource allocation may be done at this stage. For example, buffers can be dynamically allocated from a shared pool and thus optimize the delay performance in a resource-effective manner. However, this activity is probably implementation-specific and may not be subject to standardization.

Instead of accepting and routing a call through the node concerned, the call management function of a node may decide to refuse a call, based on the fact that a certain congestion threshold has been reached. Again, the criteria for congestion and call refusal may be implementation-dependent.

During the call release phase, the call management function has to free any assigned DLCI, release any resources that may have been allocated, and inform the preceding node for the call about the call release.

Upon notification from the route management function about the unavailability of certain route(s), the call management function will take appropriate action on those calls that are affected. Such action may include the release of the calls, reestablishment via alternative routes, or redistribution of traffic over available routes. Details on the need for standardization of call management procedures are for further study.

4. *Flow management* - The task of the flow management function is to control the flow of traffic on the different routes through a node. It achieves this by monitoring and measuring the utilization of different resources. Frame discard may be exercised when a certain congestion threshold is reached. Mechanisms for congestion avoidance and throughput enforcement may also be needed.

It may be necessary for the flow management function of different nodes to exchange traffic control signals. This is for further study.

Testing and Maintenance Functions - These include such activities as testing a link before being put in use, detection of link performance degradation,

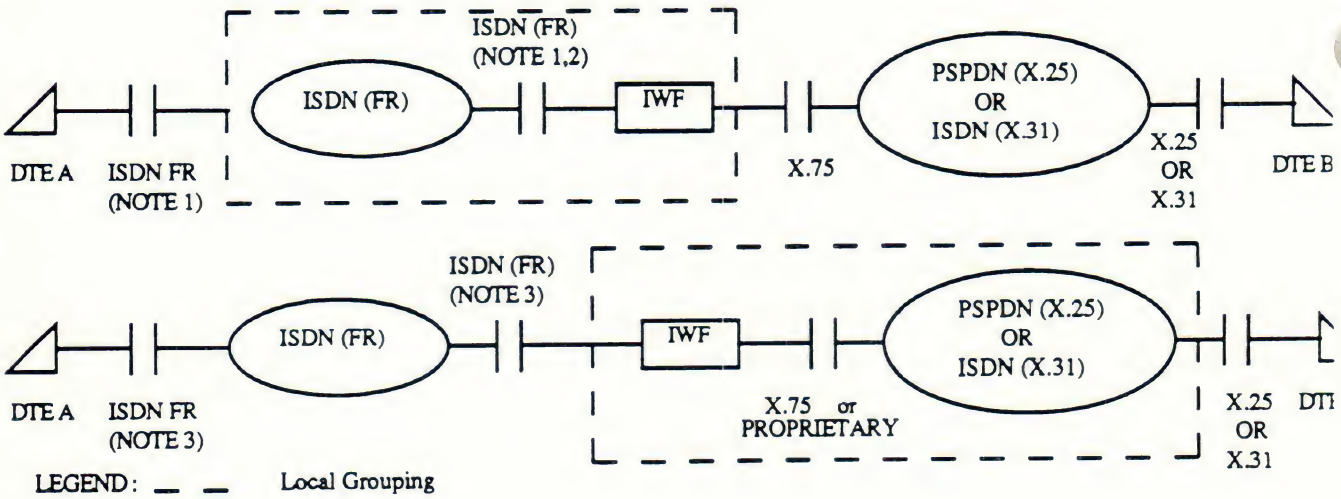
fault sectionalization, and congestion notification. Further elaboration on this subject will to a large extent depend on the details of the network management functions.

Interworking between ISDN Offering Frame Relaying and an ISDN or a PSPDN Providing Service Based on X.25

- A high level description of interworking arrangements is as follows:

1. *Possible Scenarios:* Figure 4 shows the interworking arrangements considered. When the interworking function IWF logically belongs to the ISDN (FR), interworking based on call control mapping takes place. In the case where the IWF logically belongs to the PSPDN (X.25) or ISDN (X.31), interworking based on either call control mapping or port access is possible. As shown in Figure 4, different interfaces can be specified for the different reference points, depending on whether the IWF logically belongs to the ISDN (FR), or to the PSPDN (X.25) or ISDN (X.31).

2. *IWF Logically Belonging to ISDN (FR):* TO enable interworking, ISDN (FR) in conjunction with an IWF should provide full support of the OSI network-layer service. The association of an ISDN (FR) with an IWF in such a manner could therefore be considered globally as a Type I subnetwork, in the sense defined in Recommendation X.300.



NOTES:
(1) To achieve functional compatibility, additional procedures may be required by DTE A in the U-Plane. These procedures terminate at the IWF, and are mapped into X.75 procedures.
(2) In the C-Plane, SS7, Q.931 with appropriate extensions, or proprietary protocol with equivalent functions, may be used.
(3) Additional procedures may be required in the U-Plane in the case of internetworking based on port access. These procedures terminate at the IWF, and are mapped into the X.25 procedures.

FIGURE 4 Interworking between ISDN(FR) and PSPDN(X.25) or ISDN(X.31)

A PSPDN (X.25) or an ISDN (X.31) is a Type 1 subnetwork.
As specified in X.300, the interworking arrangement between two Type I

subnetworks should be based on CCITT Recommendation X.75.

3. *IWF Logically Belonging to PSPDN (X.25)/ISDN (X.31):* The association of a PSPDN (X.25)/ISDN (X.31) with an IWF together behaves like a user terminal requesting frame-relaying service from an ISDN (FR). Therefore, the interworking arrangement can be based on ISDN (FR).

In this arrangement, interworking based on either call control mapping or port access is possible. When the port access method is used, existing call control procedures in X.25 are used for the control of virtual circuits.

Support of Existing Terminals

Terminal adapter functions should be provided that allow existing terminals to access the frame-relaying bearer service.

Interworking with Circuit-Mode Services

Configurations for interworking with other services (eg., with a CSPDN, or between different bearer services within an ISDN) may also need to be considered and are for further study.

Support of OSI Connection-Oriented Network-Layer Service

To support OSI network-layer service when the bearer service used is frame-relaying, the use of additional end-system functionality may be required and end-to-end (ie., TE-to-TE or TE-to-IWF) compatibility must be ensured.

Network-layer service could be provided through enhancements to CCITT Recommendation Q.931, and the addition of: (1) additional end-system functionality, or (2) enhancements to CCITT Recommendation Q.921 (ANSI T1.60201989) functions.

C-Plane Enhancements

Enhancements to CCITT Recommendation Q.931 are needed so that the OSI Network Service parameters can be paired with messages and information elements for all bearer services as described in CCITT Recommendation Q.931. Several enhancements to Q.931 are needed to convey all Connection Establishment and Release Primitives and Parameters in relevant Q.931 protocol elements.

U-Plane Enhancements

There are two different approaches for the mapping of OSI Data Transfer Primitives into protocol elements: (1) Layer-3 protocol elements supported by a DTE-specific protocol that is transparent for the network (preferably X.25 packet-layer procedures), and (2) protocol elements from CCITT Recommendation Q.921 (ANSI T1.602-1989) enhanced to map directly into the OSI network-layer service Data Transfer Primitives. Further study is

required for the selection and detailed definition of one of the two options.

Core Aspects of Frame Protocol for use with Frame Relay Bearer Service

Scope

The interface standard provides a description of a minimal set of data protocols to support frame relay packetized data transfer as defined in ANSI T1.6fr "Digital Signaling Specification No.1 (DSS1) - Signalling specification for Frame Relay" and ANSI T1.606 "Frame Relaying Bearer Service - Architectural Framework and Service Description" including Addendum 1 of this document.

Application

The protocol is intended to support multiple simultaneous end-user protocols within a single physical channel and is intended to be transparent to these protocols. The applicability of this standard is for frame relay bearer service.

Frame Relay Data Transfer Protocol

The core functions of the LAPD protocol that are used to provided this function are :

(1) frame delimiting, alignment, and transparency provided by the use of HDLC flags and zero bit insertion;

(2) frame multiplexing/demultiplexing using the address field;

(3) inspection of the frame to ensure that it consists of an integer number of octets prior to zero bit insertion or following zero bit extraction;

(4) inspection of the frame to ensure that it is not too long or too short;

(5) detection of (but not recovery from) transmission errors; and

(6) congestion control functions.

Frame relay frame format

The frame relay frame format is shown in Figure 1. The fields shown in the figure are described in the following subsections.

Flag Sequence

All frames start and end with the flag sequence consisting of one 0 bit followed by six contiguous 1 bits and one 0 bit. The flag preceding the address field is defined as the opening flag. The flag following the frame

check sequence (FCS) field is defined as the closing flag. The closing flag may also serve as the opening flag of the next frame; however, receivers must be able to accommodate reception of one or more consecutive flags.

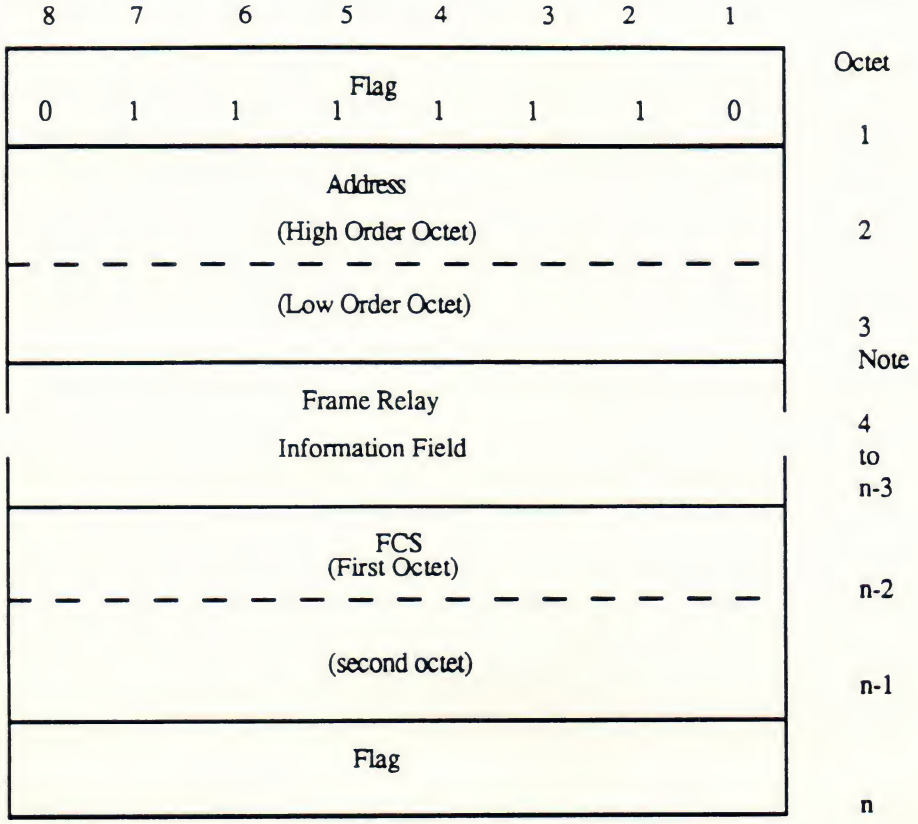


Figure 1 - Frame Relay Format with two octet address

Note - The default address field length is 2 octets.
It may be extended to either three or four octets.

Address Field

The address field consists of at least two octets and illustrated in Figure 1 but may optionally be extended up to 4 octets. The address field format is defined in para 3..

Frame Relay Information Field

The Frame Relay information field follows the address field (see 2.2) and precedes the frame check sequence (see 2.5). The contents of the user data field consists of an integer number octets.

The default maximum information field size to be supported by networks is 262 octets. All other maximum values are negotiated between users and networks and between networks. The support by networks of a negotiated

maximum value of at least 1600 octets is strongly recommended for applications such as LAN interconnect, to prevent the need for segmentation and reassembly by the user equipment.

Transparency

A transmitting data link layer entity must examine the frame content between the opening and closing flag sequences, (address, Frame Relay information and FCS fields) and must insert a 0 bit after all sequences of five contiguous 1 bits (including the last five bits of the FCS) to ensure that a flag or an abort sequence is not simulated within the frame. A receiving data link layer entity must examine the frame contents between the opening and closing flag sequences and must discard any 0 bit that directly follows five contiguous 1 bits.

Frame Checking Sequence (FCS Field

The FCS field is a 16 bit sequence. It is the ones complement of the sum (modulo 2) of :

(1) The remainder of (X^k) $(X^{15}+X^{14}+X^{13}+X^{12}+X^{11}+X^{10}+X^9+X^8+X^7+X^6+X^5+X^4+X^3+X^2+X^1+1)$ divided (modulo 2) by the generator polynomial $X^{16}+X^{12}+X^5+1$, where k is the number of bits in the frame existing between, but not including, the final bit of the opening flag and the first bit of the FCS, excluding bits inserted for transparency, and

(2) the remainder of the division (modulo 2) by the generator polynomial $X^{16}+X^{12}+X^5+1$, of the product of X^{16} by the content of the frame existing between, but not including, the final bit of the opening flag and the first bit of the FCS, excluding bits inserted for transparency.

As a typical implementation at the transmitter, the initial contents of the register of the device computing the remainder of the division is preset to all ones, and is then modified by division by the generator polynomial (as described above) on the address and frame relay information fields; the ones complement of the resulting remainder is transmitted as a 16 bit FCS sequence.

As a typical implementation at the receiver, the initial contents of the register of the device computing the remainder is preset to all ones. The final remainder after multiplication by X^{16} and then division (modulo 2) by the generator polynomial $X^{16}+X^{12}+X^5+1$ of the serial incoming protected bits and the FCS is "0001 1101 0000 1111" (X^{15} through X^0, respectively) in the absence of transmission errors.

Format Convention

Numbering Convention - The basic convention used in this section is illustrated in Figure 2. The bits are grouped into octets. The bits of an octet

are shown horizontally and are numbered from 1 to 8. Multiple octets are shown vertically and are numbered from 1 to n.

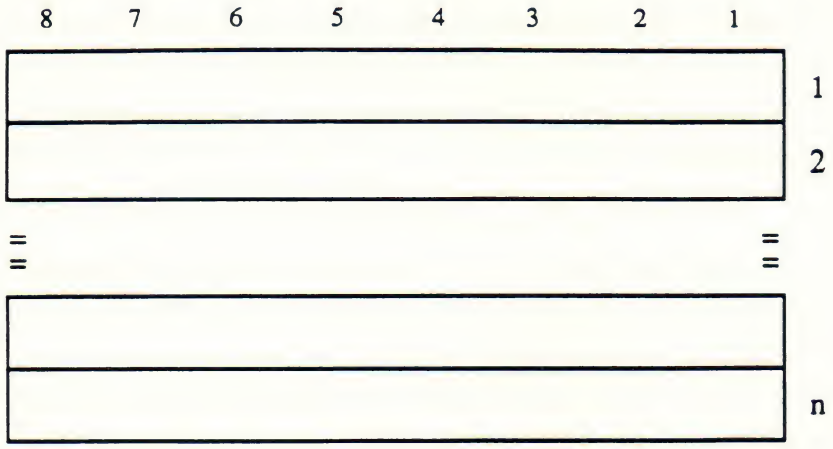


Figure 2 - Format Convention

Order of Bit Transmission - The octets are transmitted in ascending numerical order; inside an octet bit 1 is the first bit to be transmitted.

Field Mapping Convention - When a field is contained within a single octet, the lowest bit number of the field represents the lowest order value.

When a field spans more than one octet, the order of bit values progressively decreases as the octet number increases within each octet. The lowest bit number associated with the field represents the lowest order value.

For example, a bit number can be identified as a couple (o.b) where o is the octet number and b is the relative bit number within the octet. Figure 3 illustrates a field that spans from (1.3) to bit (2.7). The high order bit of the field is mapped on bit (1.3) and the low order bit is mapped on bit (2.7).

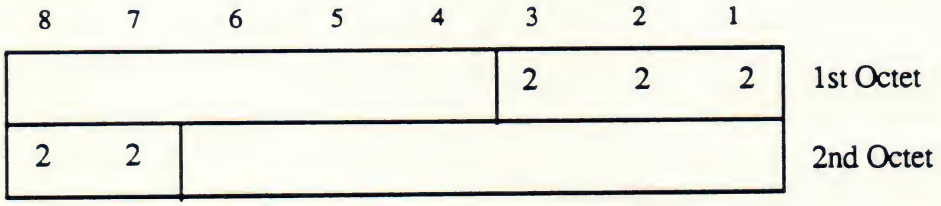


Figure 3 - Field Mapping Conventions

An exception to the preceding field mapping convention is the data link layer FCS field, which spans two octets. In this case, bit 1 of the first octet is the high order bit and bit 8 of the second octet is the low order bit (see Figure 4).

8	7	6	5	4	3	2	1	
2							2	1st Octet
2							2	2nd Octet

Figure 4 - FCS Mapping Conventions

Invalid Frames

- An invalid frame is a frame that :

(1) is not properly bounded by two flags, or
(2) has fewer than 5 octets between flags, or
(3) does not consist of an integral number of octets prior to zero bit insertion or following zero bit extraction, or
(4) contains a frame check sequence error, or
(5) contains a single octet address field, or
(6) contains a Data Link Connection Identifier (DLCI) that is not supported by the receiver.

Invalid frames are discarded without notification to the sender. No action is taken as a result of that frame.

Frame Abort - Receipt of seven or more contiguous 1 bits is interpreted as an abort and the data link layer ignores the frame currently being received.

Address Field Format

The format of the address field is shown in Figure 5. This field includes the address field extension bits, a bit reserved for use by end user equipment intended to support a Command/Response indication bit, Forward and Backward explicit congestion indicator bits, Discard eligibility Indicator, and a Data Link Identification (DLCI) field. The minimum and default length of the address field is 2 octets and it may be extended to 3 or 4 octets. To support a larger DLCI address range, the 3-octet or 4-octet address fields may be supported at the user-network interface of the network-network interface based on bilateral agreement.

Address field extension bit (EA) - The address field range is extended by reserving the first transmitted bit of the address field octets to indicate the final octet of the address field. The presence of a 1 in the first bit of the address field octet signals that it is the final octet of the address field. The two octet address field has bit one of the first octet set to a 0 and bit one of the second octet to 1.

8	7	6	5	4	3	2	1	
DLCI (high order)						C/R 0/1	EA 0	1st Octet
DLCI (low order)				FC	BC	DE	EA 1	2nd Octet

a -Address Field Format - 2 octets (default)

8	7	6	5	4	3	2	1	
DLCI (high order)						C/R 0/1	EA 0	1st Octet
DLCI (low order)				FC	BC	DE	EA 0	2nd Octet
DLCI (low order)							EA 1	3rd Octet

b -Address Field Format - 3 octets

8	7	6	5	4	3	2	1	
DLCI (high order)						C/R 0/1	EA 0	1st Octet
DLCI (low order)				FC	BC	DE	EA 0	2nd Octet
DLCI							EA 0	3rd Octet
DLCI (low order)							EA 1	

b -Address Field Format - 4 octets

FIGURE 5 - Address Format

If the extension bit is used to indicate more than two octets, then the additional octets are considered to be part of the DLCI. The support of address fields longer than two octets is an option. This option includes distinctions for supporting the address field length varying on an interface basis or on a per channel basis.

Command/Response field bit (C/R) - The C/R bit is not used by the Frame Relay protocol. The use of this field is application specific. When used C/R bit identifies a frame as either a Command or a Response. When the frame to be sent is a command frame, the C/R bit shall be set to 0. When the frame to be sent is a response frame, the C/R bit shall be set to 1.

Discard Eligibility (DE) Indicator - This bit indicates that a frame should be discarded in preference to other frames in a congestion situation, when frames must be discarded to ensure safe network operation and maintain the committed level of service within the network.

Backward Congestion (BC) Indicator - Used with source controlled transmitter rate adjustment.

Forward Congestion (FC) Indicator - Used with destination controlled transmitter rate adjustment.

Data link connection identifier (DLCI) - The DLCI is used to identify the logical connection, multiplexed within the physical channel, with which a frame is associated. All frames carried within a particular physical channel and having the same DLCI value are associated with the same logical connection.

The DLCI is an unstructured address field. For 2 octet address, bit 4 of the second octet is the least significant bit. For 3 and 4 octet addresses, bit 2 of the least octet is the least significant bit. In all cases bit 8 of the first octet is the most significant bit.

The structure of the DLCI field may be established by the network at the user to network interface or at a network to network interface subject to bilateral agreements.

DLCI values on bearer channels - In order to allow for compatibility of call control and layer management between B/H and D channels, the following ranges of DLCI's are reserved and pre-assigned. The DLCI's have local significance only.

DLCI values	FUNCTION
0	in-channel signalling
1-15	reserved
16-1007	assigned using frame relay connection procedures
1008-1022	reserved
1023	in-channel layer management

(a) - two octet address format

DLCI values	FUNCTION
0	in-channel signalling
1-2047	reserved
2048-129,023	assigned using frame relay connection procedures
129,024-131,070	reserved
131,071	in-channel layer management

(b) - 3 octet address format

TABLE 1

DLCI on the D Channel - The 6 most significant bits (bits 8 to 3) in the first octet of the address correspond to the SAPI field in ANSI standard T1.602. This DLCI subfield (bits 8-3 of first octet) values which apply on a D channel are reserved for specific functions to ensure compatibility with operation on the D channel which may also use the T1.602 protocols. A Two byte address format for T1.6ca is assumed when used on the D channel.

Note - Whether 3 or 4 octet address formats may be used on the D channel is for further study.

For Frame Relay in the D channel, only DLCI values in the range 480 - 1007 (SAPI = 32 to 62) will be assigned.

Primitives for layer-to-layer communication

General - The definition of layer-to-layer communication is in ANSI standard T1.602. A summary of the primitives supported for the Frame Relay data protocol is given in Table 2.

Generic Names - The generic names specifies the activity that should be performed. Table 2 illustrates the primitives defined for Frame Relay protocol. Note that not all primitives have associated parameters.

Congestion Control

The definition of objectives and requirements for congestion management are provided in Addendum 1 of T1.606.

Congestion in the user plane occurs when traffic arriving at a resource exceeds the network engineered level of capability. It can also occur for other reasons (eg. equipment failure). Network congestions effects the throughput rate, delay and delivery of frames to the end user.

Generic Name	Type				Parameter		Message Unit contents
	req	ind	resp	conf	priority indicator	message unit	
M- - - - - - - -L2 (cor)							
MDL-ASSIGN	x	x				x	DCLI value, CEI
MDL-ASSIGN	x					x	DCLI value, CEI
MDL-ASSIGN		x	x			x	Reason
L2(cor) - - - - L1 PH-DATA	x	x	x		x	x	Data link peer-peer message
PH-ACTIVATE	x	x					
PH-DEACTIVATE		x					

Table 2 - Primitives Involved with Frame Relay Protocol

End users should reduce their offered load in the face of network congestion. Reduction of offered load by an end user may well result in an increase in the effective throughput available to the end user during congestion.

Congestion avoidance procedures and optional explicit signalling are used at the onset of congestion to minimize the effect on the network.

Congestion recovery and the associated implicit signalling due to frame discard, is used to prevent network collapse in the face of sever congestion.

Congestion avoidance and congestion recovery are effective and complementary forms of congestion control in Frame Relay networks.

Service Principles

1. From a service perspective, call setup negotiations (eg. throughput) are rate based. This means that from the standpoint of the service provided by the network in a frame relay environment, the rate at which information is offered to the network which may be expressed in a number of information units per unit of time and is fundamental to all types of traffic to be carried.

2. Reaction by the end user to the receipt of explicit congestion notifycation (ECN) is rate based and may be subject to standardization. It is noted that window mechanisms in terminals approximate rate based mechanisms and may be used to control the rate at which traffic is offered in a network.

3. Networks should utilize, and users should react to, explicit congestion notification (ie., not mandatory but highly desirable).

4. Special provisions for the treatment of Continuous Bit stream Oriented (CBO) traffic is outside the scope of the Standard. Simplex data sources which are unable to respond to explicit congestion notification (i.e., CLLM) can only be controlled by metering and discard.

5. Explicit congestion notification (ECN) shall be provided for in the U plane. Note this congestion control and assumes that management functions such as gathering of statistics on congestion (ie., when, where, why) could be accomplished outside the U-plane.

6. Protocol mechanisms are required to convey Explicit Congestion Notification.

7. The network which perceives congestion should generate congestion - notification using the appropriate congestion control protocols. When ECN is generated, it shall be sent in the appropriate directions(s). The policies for sending ECN will be different for the source control and destination control mechanisms.

8. The network(s) shall convey the backward ECN towards the source end user and the forward ECN towards the destination end user. This requires that these indications (if set) shall not be reset as they traverse the network(s) towards source and destination users.

9. The end users (eg., private networks) may generate ECNs as per point 7.

Congestion Control Mechanisms

Commonly used end-to-end protocols operate with either source controlled or destination controlled transmit mechanisms. There are two congestion control mechanisms for the Frame Relay Bearer service to provide for both of these. These mechanisms when implemented are independent and not mutually exclusive, and may be used concurrently.

Mechanism (1): For destination controlled transmitters, the Congestion Forward (CF) bit is set in the core aspects protocol. Consistent with commonly used destination controlled protocol suites (e.g. the OSI Class 4 Transport protocol operated over the OSI Connectionless Network Service),

rate adjustment is typically a function of higher layer protocols, and end user reaction is based on the state of the CF bits that are received over a period of time.

Mechanism (2): For source controlled transmitters, the Congestion Backward (CB) bit is set in the core aspects protocol in frames transported in the reverse direction (ie., toward the transmitter. Alternatively, a consolidated Link Layer Management Message (CLLM) may be generated and sent to the transmitter on a Layer Management DLCI in the U Plane. In addition, both methods may be used. Rate adjustment is typically a function of the Data Link Layer Elements of Procedure, and end user reaction is expected to be immediate when a CB bit or CLLM is received.

Network response to congestion.

The network should in principal generate explicit congestion notification using the appropriate protocol to the source end user and/or the destination end-user around Point A. All networks must transport the CF and CB indications, either unmodified or, if in congested condition, with the appropriate indication set.

Notification in the backward direction can be accomplished using either (or both) of two optional mechanisms:

(1) A CB indication is sent with reverse traffic.

When there is reverse traffic at the time congestion is noted, then the CB indication can be "piggybacked" onto the existing frame. A CLLM may be generated to convey backward ECN.

(2) Consolidated Link Layer Management Message.

This provides reverse notification for one or more DLCI within a single frame. The generation and transport of consolidated link layer management messages by a network is optional. The CLLM is sent on the layer management DLCI in the U Plane in the backward direction (ie., toward the source end user). If a network or an end user receives this message and does not implement this option, then the consolidated link layer message should be discarded.

The network cannot rely solely on the users behavior to control network congestion. Therefore the network is expected to protect itself from catastrophic congestion situations, and may do so by monitoring the throughput of each call and invoking the frame discard strategy under congestion conditions for those calls which exceed the lesser of CIR and the information rate currently available to be allocated by the network. Therefore, as congestion can occur even when the calls do not exceed their negotiated throughput (eg., during network failures), the network should discard frames in a way that assures some fairness among users.

The use of the Discard Eligibility (DE) indicator by the users and the network is optional. The DE indicator determines whether or not this frame should be discarded by the network in preference to other frames. This decision would be necessary when the network is congested, and frames must be discarded to insure safe network operation and to maintain the committed level of service within the network. When set by a network or the user (ie., at the ingress node), the value of this indicator may be determined using one or more of the following factors:

- Access rate
- Burst size
- Committed information Rate (CIR)
- Delay

The exact list, the interaction of these factors, the precise definitions, and the way the network uses these factors is for further study. The DE indicator is symmetrical and is passed across both UNI and the NNI.

User Reponse to Congestion - Congestion in the U-Plane of a Frame Relay Bearer Service occurs when traffic, arriving at a resource (for example, memory, bandwidth, processor), exceeds the network engineered level. It can also occur for other reasons (eg., equipment failure). End users should reduce their information transfer rate upon receiving implicit or explicit indication of network congestion. Reduction of information transfer rate by an end user may result in an increase in the effective throughput available to the end user during congestion. A user of the frame relaying service should implement some form of congestion-sensitive rate adjustment function that has the following characteristics (in no specific order):

- no blocking of data flow under normal conditions even when the offered load exceeds the CIR.

- reduction to a lower information transfer rate upon detection of network congestion.

- progressive return to the negotiated information transfer rate upon congestion abatement.

The end user terminal should base the detection of network congestion on implicit congestion detection schemes as well as on explicit congestion notification.

Explicit Congestion Indication - Terminals shall have the capability to receive the explicit congestion indicator generated by the network even if they are not able to act on the information.

Note - Rate reduction strategy to be defined.

Implicit Congestion Detection - For implicit congestion detection schemes, the Frame Relay network does not send any indication to the user. Implicit

congestion schemes involve certain events available in the layer 2 elements of procedures to detect the frame loss (eg., receipt of a REJECT frame, timer recovery etc.).

The intent of this scheme is to reduce the offered load to the network by the end user. Use of such reduction by users is optional.

CHAPTER 3 - THE X.25 INTERFACE

General Description of X.25 Interface

CCITT Recommendation X.25 is titled: "Interface between Data Terminal Equipment (DTE) and Data Circuit-terminating Equipment (DCE) for Terminals Operating in the Packet Mode on Public Data Networks." However, applying the concepts of the standard seven layer model, X.25 is not strictly speaking an interface. In fact, X.25 is a set of three peer protocols as follows (See Fig.1):

1. a peer protocol between Physical Level entities in the DTE and the DCE;

2. a peer protocol between Link Control Level entities in the DTE and the network node; and

3. a peer protocol between Packet-Switched Network Level entities in the DTE and the network node.

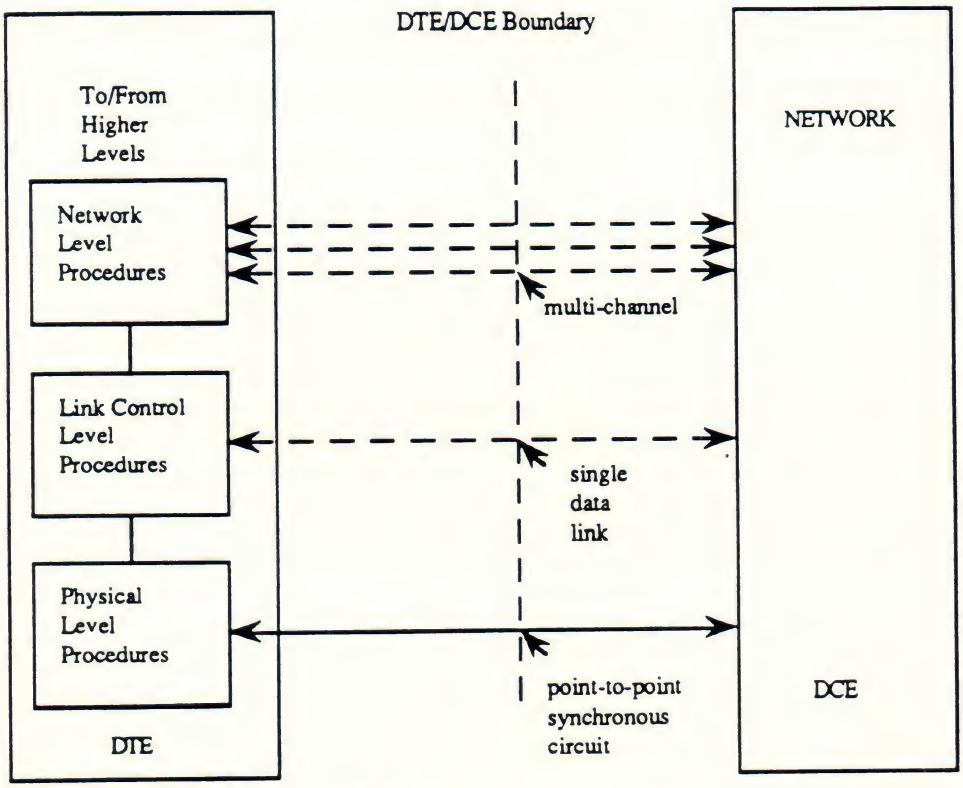


Figure 1 - Structure Of X.25

Each of these levels functions independently of the other levels, with the exception that failures at a level may affect the operation of higher levels.

The Physical Level specifies the use of a duplex, point-to-point synchronous circuit, thus providing a physical transmission path between the DTE and the Network. It also specifies the use of Recommendation V.24 (ie., the EIA

RS-232-C standard) between the DTE and a data set or modem. Therefore, no changes to the communications hardware of the DTE are required. The Physical Level also specifies the use of Recommendation X.21 through this capability is not yet widely available.

The Link Control Level specifies the use of data link control procedures which are compatible with HDLC and with ADCCP. The Link Control Level uses the principles of an ISO Class Procedures for a point-to-point balanced system; in X.25, these procedures are referred to as the Balanced Link Access Procedures (LAPB). The use of this data link control procedure ensures that packets provided by the Packet-Switched Network Level and contained in JDLC information frames (See Fig.2) are accurately exchanged between the DTE and the Network. The functions performed by the Link Control Level include :

1. the transfer of data in an efficient and timely fashion;
2. the synchronization of the link to ensure that the receiver is in step with the transmitter;
3. the detection of transmission errors and recovery from such errors; and
4. the identification and reporting of procedural errors to higher layers for recovery.

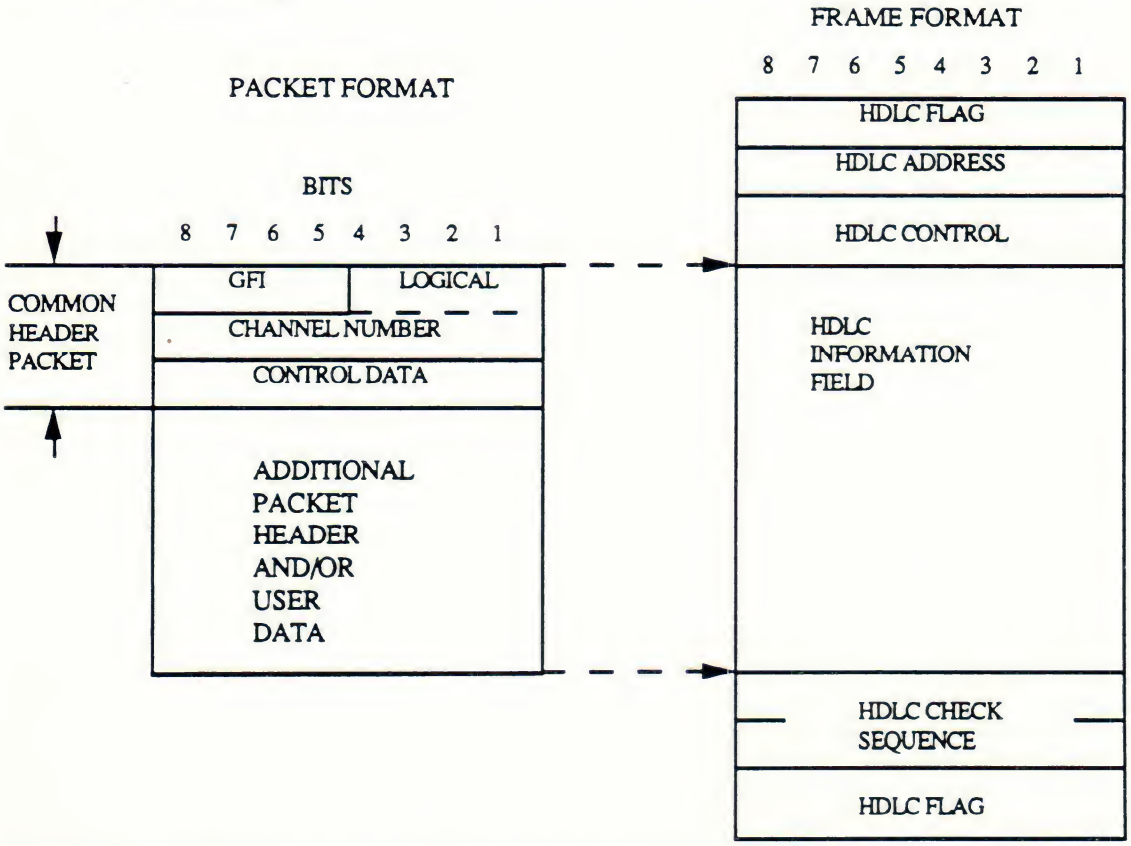


Figure 2 - General X.25 Packet and Frame Formats

The major significance of the Link Control Level is that it provides the Packet-Switched Network Level with an error-free, variable delay link between the DTE and the Network. The Packet-Switched Network Level is the highest level in X.25 and specifies the manner in which control information and user data are structured into Network Protocol Data Units called packets. The control information, including addressing information, is contained in the packet header field and allows the network to identify the DTE for which the packet is destined. It also allows a single physical circuit to support communications to numerous other DTEs concurrently.

The characteristics of the Packet-Switched Network Level Peer Protocol are further described in Section III.

Packet-Switched Network Level Services Available to X.25 DTEs

Recommendation X.25 defines a set of those peer protocols to be used between the packet-mode DTE and the common carrier equipment, generally referred to as the DCE. The X.25 Recommendation provides access to the following Network services that may be provided on public data networks:

1. switched virtual circuits (SVCs), also called virtual calls;
2. permanent virtual circuits (PVCs); and
3. datagrams

A virtual circuit (VC) is a bidirectional transparent, flow-controlled path between a pair of logical or physical ports. A switched virtual circuit is a temporary association between two DTEs and is initiated by a DTE signaling a call request to the network. A permanent virtual circuit is a permanent association existing between two DTEs which does not require call setup or call clearing action by the DTE.

A datagram (DG) is a self-contained user data unit containing sufficient information to be routed to the destination DTE (independently of all other data units) without the need for a call to be established. At this time, the datagram service is not provided on any PDNs. This service is described in the next chapter.

Multiplexing at the X.25 Interface

In order to allow a DTE to establish concurrent virtual circuits with a number of DTEs over a single physical access circuit, the X.25 Packet-Switched Network Level employs packet-interleaved Statistical Multiplexing. This multiplexing technique is used to exploit the fact that a typical virtual circuit to a remote DTE may actually be carrying data for only a small percentage of the time. Each packet contains a logical channel number which identifies the packet with a switched or permanent virtual circuit, for both directions of transmission.

A logical channel is a conceptual access path between a DTE and the

network. A logical channel, when not in use, can be dynamically assigned for a new call either originated by the local DTE or by a remote DTE. A logical channel, assigned to a call, is busy until the call is cleared. Logical channels can be viewed as analogous to dial ports in a conventional timesharing network.

Every packet consists of a three octet common packet header field as shown in Fig. 2.

Establishing and Clearing a Virtual Circuit

A signaling method is provided to allow a DTE to establish switched virtual circuits to other DTEs, using logical channel numbers at each end to locally designate these switched virtual circuits.

A DTE initiates a call by sending a CALL REQUEST packet, Fig. 3, to the Network. The CALL REQUEST packet includes the logical channel number chosen by the DTE to be used to identify all packets associated with the call. It also includes the network address of the called DTE. A facility field is present only when the DTE wishes to request an optional user facility requiring some indication at call setup. Reverse charging is an example of such a facility. User data may follow the facility field and may contain up to a maximum of 16 octets.

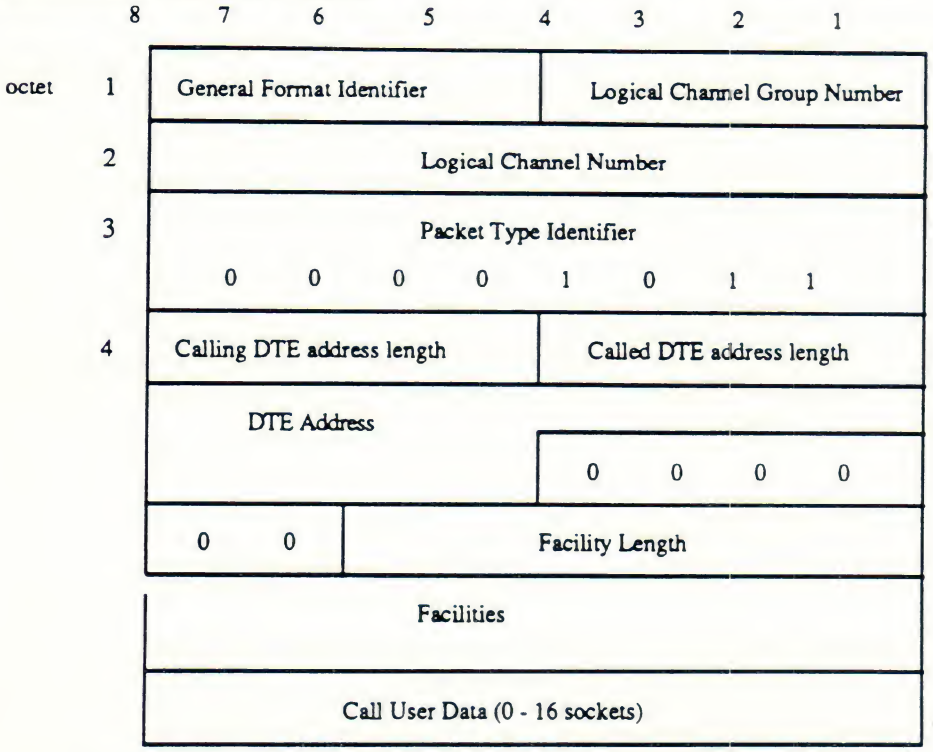


Figure 3 - CALL REQUEST and INCOMING CALL packet format

The calling DTE will receive a CALL CONNECTED packet as a response indicating that the called DTE has accepted the call (Fig 4).

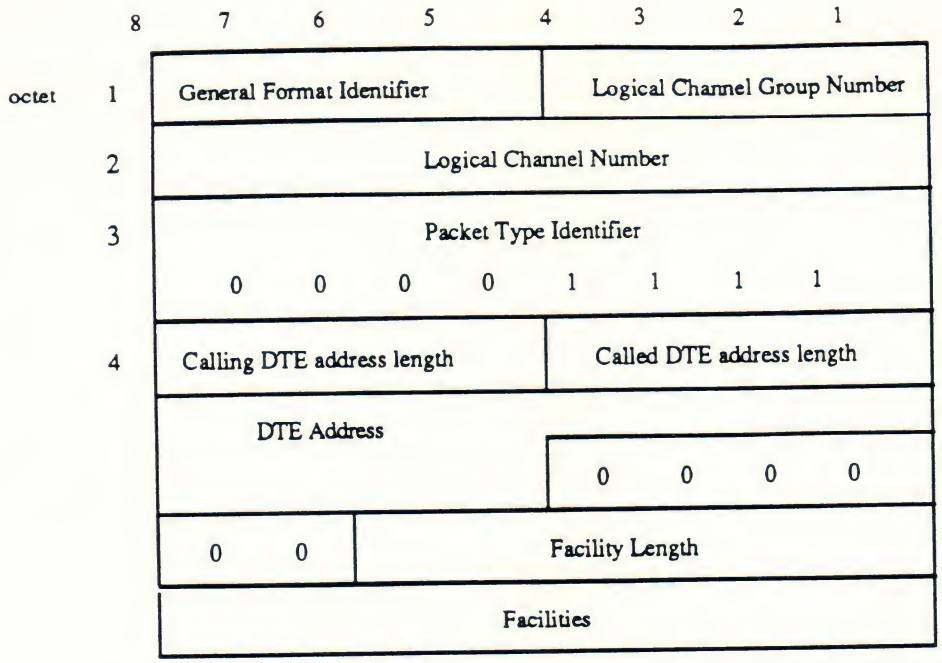


Figure 4 - CALL ACCEPTED and CALL CONNECTED packet format

If the call is refused by the called DTE or if the attempt fails, the calling DTE with receive a CLEAR INDICATION (Fig.5) indicating the appropriate call progress signal, and a one octet diagnostic field, generated by the DTE and by the network in the former and latter cases, respectively.

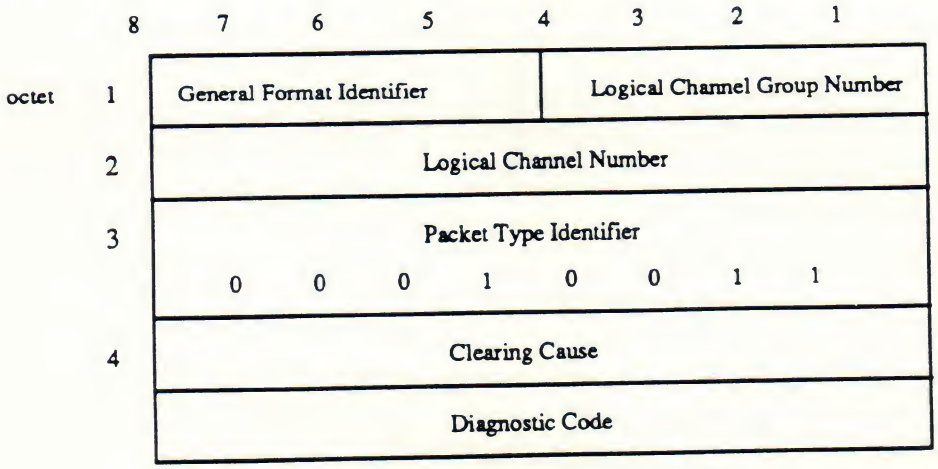


Figure 5 - CLEAR REQUEST and CLEAR INDICATION packet format

Call clearing once the call enters the data phase, may be initiated by either DTE (or by the network in the case of failure).

In any event, the elogical channel number can be used again for another call when the clearing procedure is completed, normally by the transfer of a CLEAR CONFIRMATION packet. The CLEAR CONFIRMATION packet is three octets long and identifies the logical channel for which the clear

procedures is completed.

Figure 6 illustrates the signaling and states associated with call setup and clearing on a particular logical channel.

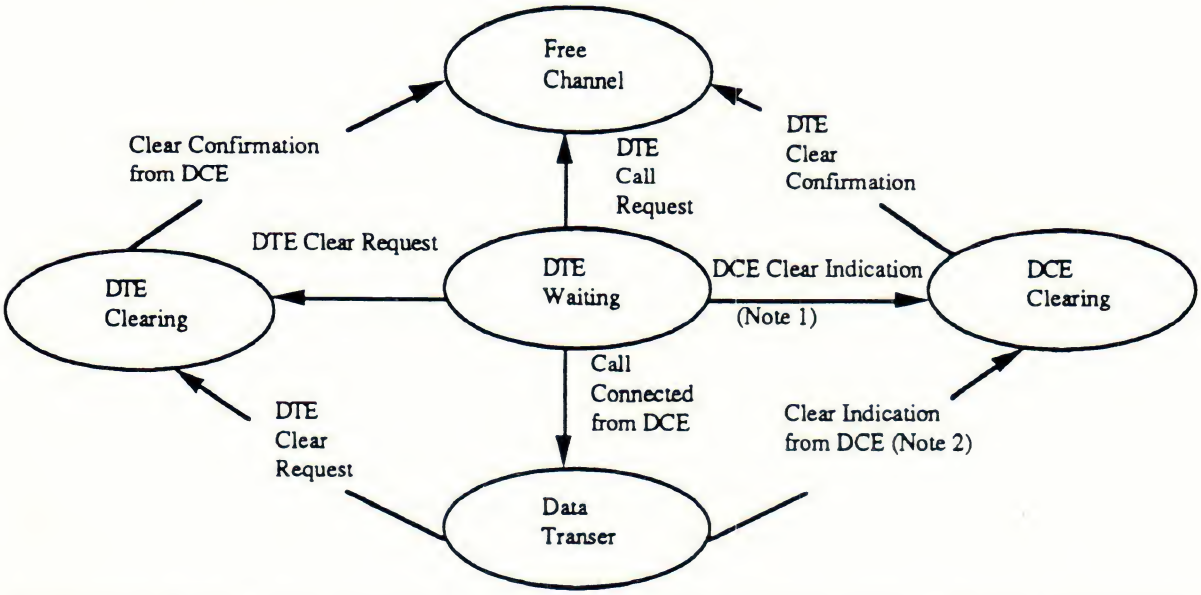


Illustration of call establishment and clearing over a logical channel Note 1: Either called DTE refused or call attempt has failed. Note 2: Either called DTE cleared down call or call cleared due to network failure

Figure 6 - CALL SETUP and CLEARING states

Data Transfer

Data packets, illustrated in Fig.7, can only be transferred across a logical channel, after the virtual circuit has been established and if flow control constraints are not violated.

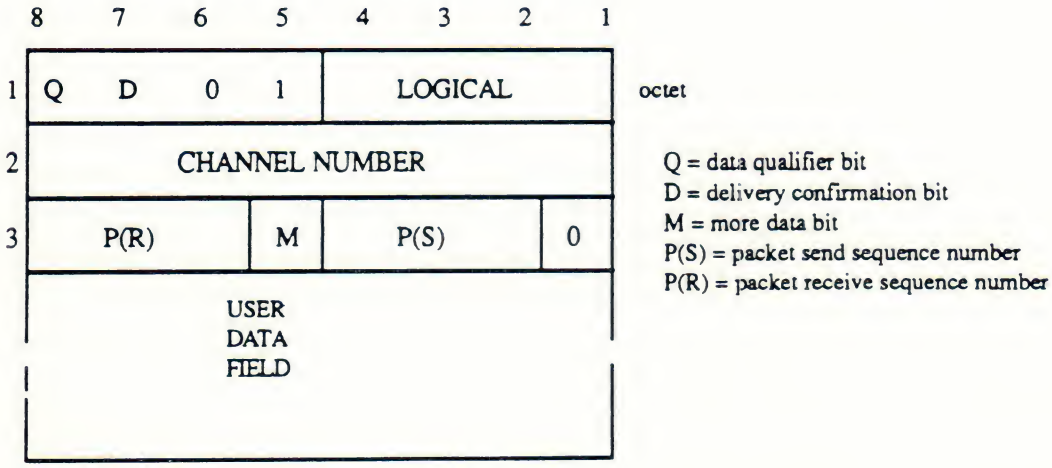


Figure 7 - DATA packet format

The data field of a DATA packet may be any length up to some maximum value. The latter may be established independently at each end of a virtual circuit. Every network will support a maximum value of 128 octets.

In order to allow two communicating DTEs to each operate at their locally selected packet sizes, the user may indicate, in a full DATA packet or any DATA packet with the D bit set to one, and there is a logical continuation of his date in the next DATA packet on a particular logical channel. This is done with the More Data "M" bit contained in the DATA packet header as indicated in Fig. 7. Only a full DATA packet may have a More Data indication since a partially full packet is treated as if it had the M bit off. Table IV defines the network treatment of DATA packets with various settings of the M and D bits.

INTERRUPT packets (Fig.8), on the other hand, may be transmitted by the DTE even when DATA packets are being flow controlled. They contain neither send nor receive sequence numbers. Only one unconfirmed

INTERRUPT may be outstanding at a given time.

octet								
1	General Format Identifier						Logical Channel Group Number	
2	Logical Channel Number							
3	Packet Type Identifier							
	0	0	1	0	0	0	1	1
4	Interrupt User Data							

Figure 8 - INTERRUPT packet format

Error Recovery

1. Reset Procedure

The reset procedure is used to reinitialize the flow control procedure on a given logical channel to the state it was in when the virtual circuit was established (i.e., all sequence numbers equal to zero and no data in transit). To reach this state, all DATA and INTERRUPT packets which may be in transit at the time of resetting are discarded. RESET REQUEST and RESET CONFIRMATION packets are used in the reset procedure.

2. Error Handling

Recommendation X.25 (1976) laid the groundwork for further study on how packet level errors were to be handled at the X.25 interface. The following principles were established:

1. procedural errors during call establishment and clearing are reported to the DTE by clearing the call;
2. procedural errors during the data transfer phase are reported to the DTE by resetting the VC;
3. a diagnostic field is included in the reset packet to provide additional information to the DTE.
4. timers are essential in resolving some deadlock conditions;
5. some DTE procedural errors are a result of the DTE and DCE not being aligned as to the subscription options provided at the interface; and
6. rudimentary error tables define the action of the DCE on receiving various packet types in various states of the interface.

Flow Control Parameter Selection

Flow Control Parameter Selection is an optional user facility agreed to for a period of time which can be used by a DTE for its logical channels. The flow control parameters considered are the packet and window sizes for each logical channel for each direction of data transmission.

When the DTE has subscribed to the facility, it may, in a CALL REQUEST packet, separately request packet sizes and window sizes for each direction of data transmission. The maximum packet sizes that may be supported on public data networks are 16, 32, 64, 128, 256, 512, and 1024 octets. If a particular packet or window size is not explicitly requested, the DCE assumes default requests of 128 octets and 2, respectively.

When the DCE transmits a CALL CONNECTED packet, it indicates in the facility field the flow control parameters to be used by the calling DTE. The only valid facility indications in the CALL CONNECTED packet as a function of the facility requests in the CALL REQUEST packet are specified by the following general negotiation rules:

1. window sizes can be changed in the direction of W = 2; and
2. packet sizes can be changed in the direction of 128 octets.

When the called DTE subscribes to the facility, the DCE transmits flow control parameter facility indications to be used by the called DTE in selecting the flow control parameters for the call. The called DTE can change the indicated values using the above negotiation rules.

The flow control parameters for logical channels used for PVCs are established at subscription time.

The network may have to constrain the available parameter ranges in order to allow the call to be established. In this case, the network is involved in the negotiations discussed above. This would occur, for example, if a requested packet size, though available domestically, was not available on a

particular international call.

Throughput Class Negotiation

Throughput Class Negotiation is an optional user facility agreed for a period of time which can be used by a DTE for virtual circuits. This facility permits negotiation on a per call basis of the throughput classes. The throughput classes are considered independently for each direction of data transmission.

A throughput class for one direction of transmission is an inherent characteristic of a virtual circuit, related to the amount of network resources allocated to it. This characteristic is meaningful when the D bit is set to zero in DATA packets. It is a measure of the throughput that is not normally exceeded on the VC. However, owing to the statistical sharing of transmission and switching resources, it is not guaranteed that the throughput class can be reached 100% or the time.

Default values are agreed between the DTE and the network. The default values correspond to the maximum throughput classes which may be associated with any virtual circuit.

A Common X.25 DTE

A common X.25 interface can be defined, which consists of the following universally available features:

- an ISO-compatible frame level procedure (i.e., LAPB);
- use of Logical Channel Number one as the starting point for logical channel assignment;
- modulo 8 packet level numbering;
- dynamic P(R) significance by use of the Delivery Confirmation bit;
- a standard procedure for selecting packet and window sizes, with defaults of 128 octets and 2, respectively;
- two mechanisms for user control data transfer (i.e., qualified DATA and INTERRUPT packets); and
- a standard way of specifying required call throughput.

CHAPTER 4 - THE NEW ORDER EMERGING

Introduction

Once off the LAN and onto the WAN, frame relay brings significant performance gains because it eliminates the processing overhead associated with packets traversing intermediate hops between packet-forwarding devices. Further savings in transmission cost will also be realized since intermediate hops are logically eliminated, thus reducing band-width requirements. Again, reducing the number of interfaces between packet-forwarding devices and T1 nodal processors to one should bring down the cost of the internetworking device. Frame relay also introduces an excellent scaling factor, which allows the user to increase connections across the wide area without spending money on high-speed data cards for the T1 nodal processor or internetworking synchronous interfaces. Thus, the cost of ownership should decrease as the network expands.

Public frame relay services from the IXC carriers and possibly from such public packet-switching vendors as US Sprint/Telenet are available today. It's a bit too early to discuss such services since specifics are not available.

A New Order Emerging

As the traffic flow moves away from nodal processor equipment toward IXC voice and internetworking devices, nodal processor vendors will find themselves in an increasingly tough position. It is unclear if the circuit-switched, time-division multiplexing (TDM) architectures can scale to support the requirements of bursty internetworking traffic while traditional circuit-switched services (i.e., voice) move back onto the public networks. Video could ultimately decide the fate of circuit switching. For example, if video traffic starts taking the form of bursty data traffic through frame-difference compression techniques, then circuit-switched architectures will be relegated to simple voice traffic.

A new switching technique that promises a more flexible way to accommodate bursty LAN-to WAN traffic combines some of the statistical efficiencies of packet switching with the low transit delays of circuit switching. Lacking a single name, this technique has been called fast packet, cell relay and asynchronous transfer mode (ATM). Whatever it's finally called, it's now being defined by CCITT and ANSI standard bodies.

With public frame relay service coming soon and existing fractional T1 services, the carriers are already providing public multiplexing and frame switching with all the built-in redundancy and rerouting necessary within the public network. Diverse access can always be provided for a price by the LECs. At the same time, the internetworking vendors deliver their datagram services to a robust public network.

If the economic trends detailed above continue, then the circuit-switched,

TDB-based T1 nodal processors may very well ultimately be relegated to customer premises T1 voice/data access to IXC services and international traffic connectivity and markets.

Efficient LAN-WAN Connection

Three issues are key to defining the model of efficient LAN-WAN interconnection: bandwidth utilization, productivity gains, and data integrity.

Bandwidth Utilization: Over the life cycle of a wide area network, telecommunications costs make up 70 to 80 percent of the total costs; equipment costs make up the rest.

Efficient bandwidth usage is critical in the LAN interconnect environment where traffic is characterized by periods of inactivity, followed by bursts of large quantities of data. In an ideal world, wide area interconnect schemes would deliver bandwidth to an application, the capacity would be made available to other applications. In the WAN arena, this is referred to as "bandwidth on demand".

Productivity Gains: The ideal interconnection would provide high-speed transmission, allowing end users to send information more quickly. Through rapid rerouting capability, end users would reliably access the wide area more frequently, thereby producing more output. There are many reasons to migrate to LAN-WAN interconnection; creating bottlenecks is not one.

Data Integrity: The globalization of business will continue, as will the attendant growth of distributed applications. These applications will be located across multiple platforms and geographically dispersed. Proprietors of distributed, heterogenous systems, such as American Airlines' Sabre reservation system and major stock exchanges, will retain customers and win new business based on their ability to deliver reliable and timely data. Efficient wide-are LAN interconnection schemes provide the optimal physical path to ensure logical integrity of enterprise relational databases.

The Frame Relay Forum

The CCITT only issues standards every four years. Although its work is essential, end users cannot wait for the formal blessing of technology that can yield their corporations greater productivity, especially when that technology holds as much near-term promise as frame relay.

To meet the challenges posed by the growing trend towards downsizing and implementing LAN-based solutions, a group of vendors formed the Frame Relay Implementors Forum. The forum is sponsored by Interop (Mountain View, Ca). It plans to enrich and expedite the international specification to

a standard that allows the interoperability of vendors' equipment. For example, a FastPacket multiplexer must operate efficiently with a CISCO (Menlo Park, Ca) or Wellfleet (Bedford, Ma) router or another vendor's bridge.
The forum comprises vendors and users, including the vendors Motorola (Shaumberg, Il) and Oracle (Redwood Shores, CA) and users FMC (Dallas, Tx) and United Stationers (Des Plaines, Il). The group began work in September 1990 when CISCO, Digital Equipment, Northern Telecom (Santa Clara, Ca), and StrataCom agreed on a Local Management Interface to help adapt the technology to commercial applications. At press time, the forum numbers approximately 45 vendors who agree to implement the specifications into their product lines.

The forum is currently working on addressing and bandwidth utilization. Multicast DLCI addresses are reserved, so a router or bridge can send a message to a predefined list of DLCIs, rather than sending the message individually and wasting bandwidth. Likewise, a global-addressing option was added to allow DLCIs on different ports to connect to the same location. Finally, an extended address was added to enlarge the header and thereby expand the possible number of addresses. This is an important issue with protocols (such as DEC-net) requiring multiple addresses of broadcast addressing capabilities.

Because multiplexers with fast packet capability can supply bandwidth on demand, and routers cannot do so, a close linkage between the multiplexer and router/bridge manufacturers is needed. Their network management systems should be integrated and allow the immediate communications of congestion information between the two devices. Unfortunately, the multiplexer vendors' proprietary or CMIP-based network management systems are not linked in any way to the router/bridge makers' SNMP-based solutions.

Implementation Considerations

Many of the Frame Relay Forum's recommendations are optional. When considering frame relay as offered by the various vendors, make sure that you understand which recommendations they have implemented. For example, have they built in the full congestion control as specified by the standard? Users with a large installed base of TCP/IP systems don't need the full suite of congestion control features offered by the frame relay standard. However, end users implementing Novell's IPX could find themselves in trouble without strong congestion control at the end devices.

Too Good To Be True?

Rarely in the information technology field does an emerging technology actually become available before the problems it was designed to solve got

out of hand. Equally unique is a three-year life cycle - from drawing board to real world implementation. Frame relay and FastPacket technology are here now. Hardware and software are available from a host of vendors, and the number is growing.

Frame relay may not be the answer for everyone. Although it combines some of the advantages of circuit and packet switching, frame relay does not beat either at its own game. Frame relay does, however, take second place in enough performance areas to earn the overall gold for LAN-WAN interconnect applications.

Is Frame Relay For You?

The needs of the application dictate the advisability of adopting the frame-relay FastPacket solution. Case Study No.1 describes a scenario where a manager might seriously consider the advantages of migrating to frame-relay FastPacket. In Cast Study No.2, the cost/benefit picture does not argue as strongly in favor of frame relay FastPacket implementation.

CASE NO.1

A large transportation company has its main office in New York, with satellite office in Cleveland, Orlando, Dallas, Chicago, and New Orleans. In each remote site, individual users access large databases, both on their local and the corporate mainframe, throughout the day. In the remote offices, LANs are the prevalent method of terminal-to-local mainframe connection. The corporate mainframe is accessed by special terminals and dial access lines. Because of expansion and regulatory changes, users at one remote site now require information from other remote sites to accurately track the shipment of goods from one region to another.

Here, the use of frame-relay technology and a wide area network backbone should be seriously considered. Installing routers with a frame-relay interface between a frame-relay interface between the LANs and a meshed WAN will give a user at any remote site real-time access to the large databases at all other sites, as well as access to the accounting information on the mainframe. This configuration will easily handle any bursts of data requests or large file transfers. Using frame relay will remove the need to have specialized terminals to connect to the mainframe as this can now be accomplished over the LAN. The need for dial lines will disappear as the WAN can now handle all aspects of the interconnection.

CASE NO.2

A chain of convenience stores with locations distributed across two states wants to connect its stores to a central site to track sales activity and product stock. Currently, an employee at each store calls the central

distribution center daily and manually transfers the day's stock status information and sales totals. To automate this process, the corporation is installing point of sale devices in each store, which automatically record the sales and inventory activity. This data is transmitted in batches twice daily over dial lines; the typical file size is small.

This organization is not a candidate for frame relay since the traffic for the WAN is light and the file sizes are relatively small. In this type of application, a public X.25 network solution would be more appropriate because it provides reliable and economical data transmission for widely dispersed, low utilization data communications.

Other offerings: As opposed to a frame-relay service, Stratacom Inc., the Calif.based T1 multiplexer vendor and long time leader in fast-packet technology, has recently teamed up with Telenet Communications Corp., the Preston, Va. based international value-added network to offer potential customers the opportunity to build their own seamless voice-data fast packet-frame relay networks.

An enhanced Stratacom IPX will be incorporated as T1 backbone nodes in Telenet's custom networks. The virtual circuit end points for the network will be Telenet's TP4/II X.25 packet switches.

The TP4/II is a router that acts as any other Telenet node allowing access via async. bisync, high-level data link control, synchronous data link control, or dial-up traffic, forwarding data to the Stratacom IPX via the CCITT recommended I.122 frame-relay interface. PBXs, however, are connected directly to the Stratacom PX>

Stratacom fast packets are fixed-length 193-bit packets that handle both voice and data. Using ADPCM and silence suppression the IPX achieves a 4:1 compression factor. With data, the fast packet network is well suited for intermittent or "bursty" traffic. Devices such as LAN gateways and X.25 bridges are ideal, since the nature of packet networks allows bandwidth on demand as opposed to an alloted circuit for each network end point.

LANS

Another area of convergence involves LANs and X.25 wide area networks. This raises the question of connectionless and connection oriented working. Lans are connectionless', because they don't involve switching or temporary paths; data simply travels along a loop or around a ring until its address matches that of the device it is passing. Wide area packet switching, on the other hand, falls under the connected-oriented label, because it consists of a network built from a series of nodes, switching to provide a route between two points.

Within a few years, connectionless working on wide area packet switches is likely, enabling full integration of LANs and WANs and bringing with it advantages such as universal network management.

Frame Relay Services Begin to Appear.

Frame relay is a form of fast packet switching that uses permanent virtual circuits upto DS1 (1.544MBit/s) speeds; chip sets that support DS3 (44.736 MBit/s) speeds are expected to be available shortly, and products thereof within a year. Frame relay's packet switching technology makes better use of bandwidth than circuit switching services because frames can be routed along the most efficient route, and frames are packed efficiently along the pipeline. Frame relay is also faster than X.25 packet switching because it eliminates most of X.25's overhead and error correction.

Frame relay makes mesh networks possible because it allows data to be sent to multiple destinations. The most economical implementation of frame relay is for mesh networks in which customers can address multiple nodes over the same access line.

- IBM is ready with a frame relay interface for the 3745 front-end processor, or a even a new FEP model. IBM is expected to provide frame relay support analogous to its existing X.25 FEP support.

- The frame relay specification moved a step closer to standard status when the American National Standards Institute (ANSI) T1S1 group approved the specification. The approval means that the packet protocol spec. has gone out to ANSI members for a letter ballot and most likely will be finalized by the organization within the next few months. After that the spec. will be sent to the CCITT for approval by that body. A key feature of the modified spec is that it includes a modified version of the local management interface (LMI). a set of messages that provide status and configuration information about the frame relay connection. (The LMI was developed by StrataCom Inc., Digital Equipment Corp., and Northern Telecom Inc.

- Automobile manufacturer Diamler-Benz AG of Stutgart, Germany, is readying a major frame relay network that will soon open for use by third parties. The network, trials on which were completed recently, comprises a mesh of 2-Mbit/s leased lines between 14 fast packet switches from StrataCom Inc. It will support a variety of protocols used in existing Diamler-Benz networks as well as frame relay services for external use. This project is seen as an important endorsement of frame relay technology in Europe

- Telematics International Inc.'s Series 9000 FRX (Frame Relay Exchange) is a statistical multiplexing network switch that uses the frame relay method of fast packet switching.

- The S9000 backbone switch for WAN's offers transfer rates of 2500 to 10000 packets per second. It supports interfaces for the company's Net25 Programmable Communications Processor, Digital Exchange, and Access Communications Processor products. The switch can be configured for 4 to 16 channels with line speeds of upto 10 Mbit/s. The S9000 deploys a proprietary implementation of frame relay. The company says its switch will comply with the frame relay interface standard now being developed by the ANSI T1S1.1 committee.

Given its strengths, frame relay may well be the dominant WAN force in coming years, allowing users to abandon the dedicated, point-to-point links that dominate today's primitive LAN interconnects. Frame relay provides a multiplexed interface between internetworking devices and T1 nodal processors in which 64 to 16,384 permanent virtual channels (depending on either LAPB [Link Access Procedure-Balanced] or LAPD [Link Access Procedured] addressing) and a control channel can be routed across the wide-area net. The T1 nodal processor delivers dynamic routing with bandwidth on demand between internetworking devices, thus creating a sophisticated logical mesh topology between packet-forwarding equipment. Every internetworking device tied into the wide area via the frame relay interface appears to have a physical network link to every other similar device; every internetworking device is now logically adjacent.

Bibliography

1. SCHWARTZ, M.: *Computer-Communication Network Design and Analysis.* Englewood Cliffs, NJ.: Prentice-Hall, 1988.

2. DEASINGTON, R.J.: *X.25 Explained protocols for packet switching and networks*, New York, NJ.: John Wiley & Sons, 1988.

3. CORBATIS, C & WILDERS, S.: *Frame Relay - containerized shipping for data.* Business Communications Review Magazine, June 1988

4. FRANK, J.: *Enterprises Network Strategies*, Gartner Group Inc. , 1989.

5. HUNTER, P.: *Team Approach to X.25 Game*, Communications Management Magazine, October 1989.

6. LIPPIS, N.: *Frame Relay Redraws the Map for Wide Area Networks*, Data Communications Magazine, July 1990.

7. HENDERSON, F.: *Frame Relay - Less is Faster*, LAN Magazine, July 1991.

8. ANSI T1.606.: *ISDN - Architectural Framework and service Description for Frame-Relaying Bearer Service*, 1990.

9. CHERUKURI, R.: *DSS1 - Core Aspects of Frame Protocol for use with frame relay bearer service*, May 1990.

Appendix B

Date: 25 Jul 88 02:59:42 GMT
From: aramis.rutgers.edu!hedrick@rutgers.edu (Charles Hedrick)
Organization: Rutgers Univ., New Brunswick, N.J.
Subject: intro to tcp admin [in 3 parts]
ssage-Id: <Jul.24.22.59.40.1988.1936@aramis.rutgers.edu>
To: tcp-ip@sri-nic.arpa

Introduction to Administration of an Internet-based Local Network

C R

C S
Computer Science Facilities Group
C I

L S

RUTGERS
The State University of New Jersey
Center for Computers and Information Services
Laboratory for Computer Science Research

24 July 1988

This is an introduction for people who intend to set up or administer a network based on the Internet networking protocols (TCP/IP).

Table of Contents

This document is intended to help people who planning to set up a new network based on the Internet protocols, or to administer an existing one. It assumes a basic familiarity with the TCP/IP protocols, particularly the structure of Internet addresses. A companion paper, "Introduction to the Internet Protocols", may provide a convenient introduction. This document does not attempt to replace technical documentation for your specific TCP/IP implementation. Rather, it attempts to give overall background that is not specific to any particular implementation. It is directed specifically at networks of "medium" complexity. That is, it is probably appropriate for a network involving several dozen buildings. Those planning to manage larger networks will need more preparation than you can get by reading this document.

In a number of cases, commands and output from Berkeley Unix are shown. Most computer systems have commands that are similar in function to these. It seemed more useful to give some actual examples that to limit myself to general talk, even if the actual output you see is slightly different.

1. The problem

This document will emphasize primarily "logical" network architecture. There are many documents and articles in the trade press that discuss actual network media, such as Ethernet, Token Ring, etc. What is generally not made clear in these articles is that the choice of network media is generally not all that critical for the overall design of a network. What can be done by the network is generally determined more by the network protocols supported, and the quality of the implementations. In practice, media are normally chosen based on purely pragmatic grounds: what media are supported by the particular types of computer that you have to connect. Generally this means that Ethernet is used for medium-scale systems, Ethernet or a network based on twisted-pair wiring for micro networks, and specialized high-speed networks (typically token ring) for campus-wide backbones, and for local networks involving super-computer and other very high-performance applications.

Thus this document assumes that you have chosen and installed individual networks such as Ethernet or token ring, and your vendor has helped you connect your computers to these network. You are now faced with the interrelated problems of

- configuring the software on your computers
- finding a way to connect individual Ethernets, token rings, etc., to form a single coherent network
- connecting your networks to the outside world

My primary thesis in this document is that these decisions require a bit of advance thought. In fact, most networks need an

"architecture". This consists of a way of assigning addresses, a way of doing routing, and various choices about how hosts interact with the network. These decisions need to be made for the entire network, preferably when it is first being installed.

2. Routing and Addressing

Many of the decisions that you need to make in setting up TCP/IP depend upon routing, so it will be best to give a bit of background on that topic now. I will return to routing in a later section when discussing gateways and bridges. In general, IP datagrams pass through many networks while they are going between the source and destination. Here's a typical example. (Addresses used in the examples are taken from Rutgers University.)

```
          network 1                    network 2      network 3
           128.6.4                      128.6.21       128.121
     ---------------------------------  ----------  ---------------
     |                |        |    |    |    |    |    |          |
     |                |        |    |    |    |    |    |          |
   128.6.4.2        128.6.4.3  128.6.4.1  128.6.21.1  128.121.50.2
                               128.6.21.2 128.121.50.1

   computer A       computer B  gateway R   gateway S   computer C
```

This diagram shows three normal computer systems, two gateways, and three networks. The networks might be Ethernets, token rings, or any other sort. Network 2 could even be a single point to point line connecting gateways R and S.

Note that computer A can send datagrams to computer B directly, using network 1. However it can't reach computer C directly, since they aren't on the same network. There are several ways to connect separate networks. This diagram assumes that gateways are used. (In a later section, we'll look at an alternative.) In this case, datagrams going between A and C must be sent through gateway R, network 2, and gateway S. Every computer that uses TCP/IP needs appropriate information and algorithms to allow it to know when datagrams must be sent through a gateway, and to choose an appropriate gateway.

Routing is very closely tied to the choice of addresses. Note that the address of each computer begins with the number of the network that it's attached to. Thus 128.6.4.2 and 128.6.4.3 are both on network 128.6.4. Next, notice that gateways, whose job is to connect networks, have an address on each of those networks. For example, gateway R connects networks 128.6.4 and 128.6.21. Its connection to network 128.6.4 has the address 128.6.4.1. Its connection to network 128.6.21 has the address 128.6.21.2.

Because of this association between addresses and networks, routing decisions can be based strictly on the network number of the

destination. Here's what the routing information for computer A might look like:

network	gateway	metric
128.6.4	none	0
128.6.21	128.6.4.1	1
128.121	128.6.4.1	2

From this table, computer A can tell that datagrams for computers on network 128.6.4 can be sent directly, and datagrams for computers on networks 128.6.21 and 128.121 need to be sent to gateway R for forwarding. The "metric" is used by some routing algorithms as a measure of how far away the destination is. In this case, the metric simply indicates how many gateways the datagram has to go through. (This is often referred to as a "hop count".)

When computer A is ready to send a datagram, it examines the destination address. The network number is taken from the beginning of the address and looked up in the routing table. The table entry indicates whether the packet should be sent directly to the destination or to a gateway.

Note that a gateway is simply a computer that is connected to two different networks, and is prepared to forward packets between them. In many cases it is most efficient to use special-purpose equipment designed for use as a gateway. However it is perfectly possible to use ordinary computers as gateways, as long as they have more than one network interface, and their software is prepared to forward datagrams. Most major TCP/IP implementations (even for microcomputers) are designed to let you use your computer as a gateway. However some of this software has limitations that can cause trouble for your network.

3. Choosing an addressing structure

The first comment to make about addresses is a warning: Before you start using a TCP/IP network, you must get one or more official network numbers. TCP/IP addresses look like this: 128.6.4.3. This address is used by one computer at Rutgers University. The first part of it, 128.6, is a network number, allocated to Rutgers by a central authority. Before you start allocating addresses to your computers, you must get an official network number. Unfortunately, some people set up networks using either a randomly-chosen number, or a number taken from examples in vendor documentation. While this may work in the short run, it is a very bad idea for the long run. Eventually, you will want to connect your network to some other organization's network. Even if your organization is highly secret and very concerned about security, somewhere in your organization there is going to be a research computer that ends up being connected to a nearby university. That university will probably be connected to a large-scale national network. As soon as one of your datagrams

escapes your local network, the organization you are talking to is going to become very confused, because the addresses that appear in your datagrams are probably officially allocated to someone else.

The solution to this is simple: get your own network number from the beginning. It costs nothing. If you delay it, then sometime years from now you are going to be faced with the job of changing every address on a large network. Network numbers are currently assigned by the DDN Network Information Center, SRI International, 333 Ravenswood Avenue, Menlo Park, California 94025 (telephone: 800-235-3155). You can get a network number no matter what your network is being used for. You do not need authorization to connect to the Defense Data Network in order to get a number. The main piece of information that will be needed when you apply for a network number is that address class that you want. See below for a discussion of this.

In many ways, the most important decision you have to make in setting up a network is how you will assign Internet addresses to your computers. This choice should be made with a view of how your network is likely to grow. Otherwise, you will find that you have to change addresses. When you have several hundred computers, address changes can be nearly impossible.

Addresses are critical because Internet datagrams are routed on the basis of their address. For example, addresses at Rutgers University have a 2-level structure. A typical address is 128.6.4.3. 128.6 is assigned to Rutgers University by a central authority. As far as the outside world is concerned, 128.6 is a single network. Other universities send any packet whose address begins with 128.6 to the nearest Rutgers gateway. However within Rutgers, we divide up our address space into "subnets". We use the next 8 bits of address to indicate which subnet a computer belongs to. 128.6.4.3 belongs to subnet 128.6.4. Generally subnets correspond to physical networks, e.g. separate Ethernets, although as we will see later there can be exceptions. Systems inside Rutgers, unlike those outside, contain information about the Rutgers subnet structure. So once a packet for 128.6.4.3 arrives at Rutgers, the Rutgers network will route it to the departmental Ethernet, token ring, or whatever, that has been assigned subnet number 128.6.4.

When you start a network, there are several addressing decisions that face you:

- Do you subdivide your address space?
- If so, do you use subnets or class C addresses?
- You do you allocate subnets or class C networks?
- How big an address space do you need?

5.1 Should you subdivide your address space?

It is not necessary to use subnets at all. There are network technologies that allow an entire campus or company to act as a single large logical Ethernet, so that no internal routing is necessary. If you use this technology, then you do not need to subdivide your address space. In that case, the only decision you have to make is what class address to apply for. However we recommend using either a subnet approach or some other method of subdividing your address space in all cases:

- In section 5.2 we will argue that internal gateways are desirable for networks of any degree of complexity.

- Even if you do not need gateways now, you may find later that you need to use them. Thus it probably makes sense to assign addresses as if each Ethernet or token ring was going to be a separate subnet. This will allow for conversion to real subnets later if it proves necessary.

- For network maintenance purposes, it is convenient to have addresses whose structure corresponds to the structure of the network. That is, when you see a stray packet from system 128.6.4.3, it is nice to know that all addresses beginning with 128.6.4 are in a particular building.

5.2 Subnets vs. multiple network numbers

Suppose that you have been convinced that it's a good idea to impose some structure on your addresses. The next question is what that structure should be. There are two basic approaches. One is subnets. The other is multiple network numbers.

The Internet standards specify what constitutes a network number. For numbers beginning with 128 through 191 (the most common numbers these days), the first two octets form the network number. E.g. in 140.3.50.1, 140.3 is the network number. Network numbers are assigned to a particular organization. What you do with the next two octets is up to you. You could choose to make the next octet be a subnet number, or you could use some other scheme entirely. Gateways within your organization will be set up to know the subnetting scheme that you are using. However outside your organization, no one will know that 140.3.50 is one subnet and 140.3.51 is another. They will simply know that 140.3 is your organization. Unfortunately, this ability to add additional structure to the address via subnets was not present in the original TCP/IP specifications. Thus some software is incapable of being told about subnets.

If enough of the software that you are using has this problem, it may be impractical for you to use subnets. Some organizations have used a different approach. It is possible for an organization to apply for

several network numbers. Instead of dividing a single network number, say 140.3, into several subnets, e.g. 140.3.1 through 140.3.10, you could apply for 10 different network numbers. Thus you might be assigned the range 140.3 through 140.12. All TCP/IP software will know that these are different network numbers.

While using separate network numbers will work just fine within your organization, it has two very serious disadvantages. The first, and less serious, is that it wastes address space. There are only about 16,000 possible class B addresses. We cannot afford to waste 10 of them on your organization, unless it is very large. This objection is less serious because you would normally ask for class C addresses for this purpose, and there are about 2 million possible class C addresses.

The more serious problem with using several network numbers rather than subnets is that it overloads the routing tables in the rest of the Internet. As mentioned above, when you divide your network number into subnets, this division is known within your organization, but not outside it. Thus systems outside your organization need only one entry in their tables in order to be able to reach you. E.g. other universities have entries in their routing tables for 128.6, which is the Rutgers network number. If you use a range of network numbers instead of subnets, that division will be visible to the entire Internet. If we used 128.6 through 128.16 instead of subdividing 128.6, other universities would need entries for each of those network numbers in their routing tables. As of this writing the routing tables in many of the national networks are exceeding the size of the current routing technology. It would be considered extremely unfriendly for any organization to use more than one network number. This may not be a problem if your network is going to be completely self-contained, or if only one small piece of it will be connected to the outside world. Nevertheless, most TCP/IP experts strongly recommend that you use subnets rather than multiple networks. The only reason for considering multiple networks is to deal with software that cannot handle subnets. This was a problem a few years ago, but is currently less serious. As long as your gateways can handle subnets, you can deal with a few individual computers that cannot by using "proxy ARP" (see below).

3.3 How to allocate subnet or network numbers

Now that you have decided to use subnets or multiple network numbers, you have to decide how to allocate them. Normally this is fairly easy. Each physical network, e.g. Ethernet or token ring, is assigned a separate subnet or network number. However you do have some options.

In some cases it may make sense to assign several subnet numbers to a single physical network. At Rutgers we have a single Ethernet that spans three buildings, using repeaters. It is very clear to us that as computers are added to this Ethernet, it is going to have to be

split into several separate Ethernets. In order to avoid having to change addresses when this is done, we have allocated three different subnet numbers to this Ethernet, one per building. (This would be handy even if we didn't plan to split the Ethernet, just to help us keep track of where computers are.) However before doing this, make very sure that the software on all of your computers can handle a network that has three different network numbers on it.

You also have to choose a "subnet mask". This is used by the software on your systems to separate the subnet from the rest of the address. So far we have always assumed that the first two octets are the network number, and the next octet is the subnet number. For class B addresses, the standards specify that the first two octets are the network number. However we are free to choose the boundary between the subnet number and the rest of the address. It's very common to have a one-octet subnet number, but that's not the only possible choice. Let's look again at a class B address, e.g. 128.6.4.3. It is easy to see that if the third octet is used for a subnet number, there are 256 possible subnets and within each subnet there are 256 possible addresses. (Actually, the numbers are more like 254, since it is generally a bad idea to use 0 or 255 for subnet numbers or addresses.) Suppose you know that you will never have more than 128 computers on a given subnet, but you are afraid you might need more than 256 subnets. (For example, you might have a campus with lots of small buildings.) In that case, you could define 10 bits for the subnet number, leaving 6 bits for addresses within each subnet. This choice is expressed by a bit mask, using ones for the bits used by the network and subnet number, and 0's for the bits used for individual addresses. Our normal subnet choice is given as 255.255.255.0. If we chose 10 bit subnet numbers and 6 bit addresses, the subnet mask would be 255.255.255.192.

Generally it is possible to specify the subnet mask for each computer as part of configuring its TCP/IP software. The TCP/IP protocols also allow for computers to send a query asking what the subnet mask is. If your network supports broadcast queries, and there is at least one computer or gateway on the network that knows the subnet mask, it may be unnecessary to set it on the other computers. (This capability brings with it a whole new set of possible problems. One well-known TCP/IP implementation would answer with the wrong subnet mask when queried, thus leading causing every other computer on the network to be misconfigured.)

3.3.1 Dealing with multiple "virtual" subnets on one network

Most software is written under the assumption that every computer on the local network has the same subnet number. When traffic is being sent to a machine with a different subnet number, the software will generally expect to find a gateway to handle forwarding to that subnet. Let's look at the implications. Suppose subnets 128.6.19 and 128.6.20 are on the same Ethernet. Consider the way things look from the point of view of a computer with address 128.6.19.3. It will have

no problem sending to other machines with addresses 128.6.19.x. They are on the same subnet, and so our computer will know to send directly to them on the local Ethernet. However suppose it is asked to send a packet to 128.6.20.2. Since this is a different subnet, most software will expect to find a gateway that handles forwarding between the two subnets. Of course there isn't a gateway between subnets 128.6.19 and 128.6.20, since they are on the same Ethernet. Thus it must be possible to tell your software that 128.6.20 is actually on the same Ethernet.

For the most common TCP/IP implementations, it is possible to deal with more than one subnet on a network. For example, Berkeley Unix allows you to define gateways using a command "route add". Suppose that you get from subnet 128.6.19 to subnet 128.6.4 using a gateway whose address is 128.6.19.1. You would use the command

```
route add 128.6.4.0 128.6.19.1 1
```

This says that to reach subnet 128.6.4, traffic should be sent via the gateway at 128.6.19.1, and that the route only has to go through one gateway. The "1" is referred to as the "routing metric". If you use a metric of 0, you are saying that the destination subnet is on the same network, and no gateway is needed. In our example, on system 128.6.19.3, you would use

```
route add 128.6.20.0 128.6.19.1 0
```

The actual address used in place of 128.6.19.1 is irrelevant. The metric of 0 says that no gateway is actually going to be used, so the gateway address is not used. However it must be a legal address on the local network.

Note that the commands in this section are simply examples. You should look in the documentation for your particular implementation to see how to configure your routing.

3.4 Choosing an address class

When you apply for an official network number, you will be asked what class of network number you need. The possible answers are A, B, and C. This affects how large an address space you will be allocated. Class A addresses are one octet long, class B addresses are 2 octets, and class C addresses are 3 octets. This represents a tradeoff: there are a lot more class C addresses than class A addresses, but the class C addresses don't allow as many hosts. The idea was that there would be a few very large networks, a moderate number of medium-size ones, and a lot of mom-and-pop stores that would have small networks. Here is a table showing the distinction:

class	range of first octet	network	rest	possible addresses
A	1 - 126	p	q.r.s	16777214
B	128 - 191	p.q	r.s	65534

C 192 - 223 p.q.r s 254

for example network 10, a class A network, has addresses between 10.0.0.1 and 10.255.255.254. So it allows 254**3, or about 16 million possible addresses. (Actually, network 10 has allocated addresses where some of the octets are zero, so there are a few more networks possible.) Network 192.12.88, a class C network has hosts between 192.12.88.1 and 128.12.88.254, i.e. 254 possible hosts.

In general, you will be expected to choose the lowest class that will provide you with enough addresses to handle your growth over the next few years. In general organizations that have computers in many buildings will probably need and be able to get a class B address, assuming that they are going to use subnetting. (If you are going to use many separate network numbers, you would ask for a number of class C addresses.) Class A addresses are normally used only for large public networks and for a few very large corporate networks.

4. Setting up routing for an individual computer

All TCP/IP implementations require some configuration for each host. In some cases this is done in a "system generation". In other cases, various startup and configuration files must be set up on the system. Still other systems get configuration information across the network from a "server". While the details differ, the same kinds of information need to be supplied for most implementations. This includes

- parameters describing the specific machine, such as its Internet address.

- parameters describing the network, such as the subnet mask (if any)

- routing software and the tables that drive it

- startup of various programs needed to handle network tasks

Before a machine is installed on your network, a coordinator should assign it a host name and Internet address. If the machine has more than one network interface, you must assign a separate Internet address for each. (In most cases, the same host name can be used. The name goes with the machine as a whole, whereas the address is associated with the connection to a specific network.) The address should begin with the network number for the network to which it is to be attached. We recommend that you assign addresses starting from 1. Should you find that you need more subnets than your current subnet mask allows, you may later need to expand the subnet mask to use more bits. If all addresses use small numbers, this will be possible.

Generally the Internet address must be specified individually in a configuration file on each computer. However some computers

(particularly those without permanent disks on which configuration information could be stored) find out their Internet address by sending a broadcast request over the network. In that case, you will have to make sure that some other system is configured to answer the request. When a system asks for its Internet address, enough information must be put into the request to allow another system to recognize it and to send a response back. For Ethernet systems, generally the request will include the Ethernet address of the requesting system. Ethernet addresses are assigned by the computer manufacturers, and are guaranteed to be unique. Thus they are a good way of identifying the computer. And of course the Ethernet address is also needed in order to send the response back. If it is used as the basis for address lookup, the system that is to answer the request will need a table of Ethernet addresses and the corresponding Internet address. The only problem in constructing this table will be finding the Ethernet address for each computer. Generally, computers are designed so that they print the Ethernet address on the console shortly after being turned on. However in some cases you may have to type a command that displays information about the Ethernet interface.

Generally the subnet mask should be specified in a configuration file associated with the computer. (For Unix systems, the "ifconfig" command is used to specify both the Internet address and subnet mask.) However there are provisions in the IP protocols for a computer to broadcast a request asking for the subnet mask. The subnet mask is an attribute of the network. That is, it is the same for all computers on a given subnet. Thus there is no separate subnet table corresponding to the Ethernet/Internet address mapping table used to answer address queries. Generally any machine on the network that believes it knows the subnet mask will answer any query about the subnet mask. For that reason, an incorrect subnet mask setting on one machine can cause confusion throughout the network.

Normally the configuration files do roughly the following things:

- enable each of the network interfaces (Ethernet interface, serial lines, etc.) Normally this involves specifying an Internet address and subnet mask for each, as well as other options that will be described in your vendor's documentation.

- establish network routing information, either by commands that add fixed routes, or by starting a program that obtains them dynamically.

- turn on the name server (used for looking up names and finding the corresponding Internet address -- see the section on the domain system in the Introduction to TCP/IP).

- set various other information needed by the system software, such as the name of the system itself.

- start various "daemons". These are programs that provide network services to other systems on the network, and to users on this system.

It is not practical to document these steps in detail, since they differ for each vendor. This section will concentrate on a few issues where your choice will depend upon overall decisions about how your network is to operate. These overall network policy decisions are often not as well documented by the vendors as the details of how to start specific programs. Note that some care will be necessary to integrate commands that you add for routing, etc., into the startup sequence at the right point. Some of the most mysterious problems occur when network routing is not set up before a program needs to make a network query, or when a program attempts to look up a host name before the name server has finished loading all of the names from a master name server.

4.1 How datagrams are routed

If your system consists of a single Ethernet or similar medium, you do not need to give routing much attention. However for more complex systems, each of your machines needs a routing table that lists a gateway and interface to use for every possible destination network. A simple example of this was given at the beginning of this section. However it is now necessary to describe the way routing works in a bit more detail. On most systems, the routing table looks something like the following. (This example was taken from a system running Berkeley Unix, using the command "netstat -n -r". Some columns containing statistical information have been omitted.)

Destination	Gateway	Flags	Interface
128.6.5.3	128.6.7.1	UHGD	il0
128.6.5.21	128.6.7.1	UHGD	il0
127.0.0.1	127.0.0.1	UH	lo0
128.6.4	128.6.4.61	U	pe0
128.6.6	128.6.7.26	U	il0
128.6.7	128.6.7.26	U	il0
128.6.2	128.6.7.1	UG	il0
10	128.6.4.27	UG	pe0
128.121	128.6.4.27	UG	pe0
default	128.6.4.27	UG	pe0

The example system is connected to two Ethernets:

controller	network	address	other networks
il0	128.6.7	128.6.7.26	128.6.6
pe0	128.6.4	128.6.4.61	none

The first column shows the designation for the controller hardware that connects the computer to that Ethernet. (This system happens to have controllers from two different vendors. The first one is made by Interlan, the second by Pyramid.) The second column is the network number for the network. The third column is this computer's Internet address on that network. The last column shows other subnets that share the same physical network.

Now let's look at the routing table. For the moment, let us ignore the first 3 lines. The majority of the table consists of a set of entries describing networks. For each network, the other three columns show where to send datagrams destined for that network. If the "G" flag is present in the third column, datagrams for that network must be sent through a gateway. The second column shows the address of the gateway to be used. If the "G" flag is not present, the computer is directly connected to the network in question. So datagrams for that network should be sent using the controller shown in the third column. The "U" flag in the third column simply indicates that the route specified by that line is up, i.e. that no errors have occured indicating that the path is unusable.

The first 3 lines show "host routes", indicated by the "H" flag in column three. Routing tables normally have entries for entire networks or subnets. For example, the entry

128.6.2	128.6.7.1	UG	il0

indicates that datagrams for any computer on network 128.6.2 (i.e. addresses 128.6.2.1 through 128.6.2.254) should be sent to gateway 128.6.7.1 for forwarding. However sometimes routes apply only to a specific computer, rather than to a whole network. In that case, a host route is used. The first column then shows a complete address, and the "H" flag is present in column 3. E.g. the entry

128.6.5.21	128.6.7.1	UHGD	il0

indicates that datagrams for the specific address 128.6.5.21 should be sent to the gateway 128.6.7.1. As with network routes, the "G" flag is used for routes that involve a gateway. The "D" flag indicates that the route was added dynamically, based on an ICMP redirect message from a gateway. (See below for details.)

The following route is special:

127.0.0.1	127.0.0.1	UH	lo0

127.0.0.1 is the address of the "loopback device". This is a dummy software module. Any datagram sent out through that "device" appears immediately as input. It can be used for testing. The loopback address is also handy for sending queries to programs that are designed to respond to network queries, but happen to be running on the same computer. (Why bother to use your network to talk to a program that is on the same machine you are?)

Finally, there are "default" routes, e.g.

default	128.6.4.27	UG	pe0

This route is used for datagrams that don't match any other entry. In this case, they are sent to a gateway with address 128.6.4.27.

In most systems, datagrams are routed by looking up the destination address in a table such as the one just described. If the address

matches a specific host route, then that is used. Otherwise, if it matches a network route, that is used. If no other route works, the default is used. If there is no default, normally the user gets an error message such as "network is unreachable".

The following sections will describe several ways of setting up these routing tables. Generally, the actual operation of sending packets doesn't depend upon which method you use to set up the routes. When a packet is to be sent, its destination is looked up in the table. The different routing methods are simply more and less sophisticated ways of setting up and maintaining the tables.

4.2 Fixed routes

The simplest way of doing routing is to have your configuration contain commands to set up the routing table at startup, and then leave it alone. This method is practical for relatively small networks, particularly if they don't change very often.

Most computers automatically set up some routing entries for you. Unix will add an entry for the networks to which you are directly connected. For example, your startup file might contain the commands

```
ifconfig ie0 128.6.4.4 netmask 255.255.255.0
ifconfig ie1 128.6.5.35 netmask 255.255.255.0
```

These specify that there are two network interfaces, and your addresses on them. The system will automatically create routing table entries

128.6.4	128.6.4.4	U	ie0
128.6.5	128.6.5.35	U	ie1

These specify that datagrams for the local subnets, 128.6.4 and 128.6.5, should be sent out the corresponding interface.

In addition to these, your startup files would contain commands to define routes to whatever other networks you wanted to reach. For example,

```
route add 128.6.2.0 128.6.4.1  1
route add 128.6.6.0 128.6.5.35 0
```

These commands specify that in order to reach network 128.6.2, a gateway at address 128.6.4.1 should be used, and that network 128.6.6 is actually an additional network number for the physical network connected to interface 128.6.5.35. Some other software might use different commands for these cases. Unix differentiates them by the "metric", which is the number at the end of the command. The metric indicates how many gateways the datagram will have to go through to get to the destination. Routes with metrics of 1 or greater specify the address of the first gateway on the path. Routes with metrics of

0 indicate that no gateway is involved -- this is an additional network number for the local network.

Finally, you might define a default route, to be used for destinations not listed explicitly. This would normally show the address of a gateway that has enough information to handle all possible destinations.

If your network has only one gateway attached to it, then of course all you need is a single entry pointing to it as a default. In that case, you need not worry further about setting up routing on your hosts. (The gateway itself needs more attention, as we will see.) The following sections are intended to provide help for setting up networks where there are several different gateways.

4.3 Routing redirects

Most Internet experts recommend leaving routing decisions to the gateways. That is, it is probably a bad idea to have large fixed routing tables on each computer. The problem is that when something on the network changes, you have to go around to many computers and update the tables. If changes happen because a line goes down, service may not be restored until someone has a chance to notice the problem and change all the routing tables.

The simplest way to keep routes up to date is to depend upon a single gateway to update your routing tables. This gateway should be set as your default. (On Unix, this would mean a command such as "route add default 128.6.4.27 1", where 128.6.4.27 is the address of the gateway.) As described above, your system will send all datagrams to the default when it doesn't have any better route. At first, this strategy does not sound very good if you have more than one gateway. After all, if all you have is a single default entry, how will you ever use the other gateways in the cases where they are better? The answer is that most gateways are able to send "redirects" when they get datagrams for which there is a better route. A redirect is a specific kind of message using the ICMP (Internet Control Message Protocol). It contains information that generally translates to "In the future, to get to address XXXXX, please use gateway YYYYY instead of me". Correct TCP/IP implementations use these redirects to add entries to their routing table. Suppose your routing table starts out as follows:

Destination	Gateway	Flags	Interface
127.0.0.1	127.0.0.1	UH	lo0
128.6.4	128.6.4.61	U	pe0
default	128.6.4.27	UG	pe0

This contains an entry for the local network, 128.6.4, and a default pointing to the gateway 128.6.4.27. Suppose there is also a gateway 128.6.4.30, which is the best way to get to network 128.6.7. How do

you find it? Suppose you have datagrams to send to 128.6.7.23. The first datagram will go to the default gateway, since that's the only thing in the routing table. However the default gateway, 128.6.4.27, will notice that 128.6.4.30 would really be a better route. (How it does that is up to the gateway. However there are some fairly simple methods for a gateway to determine that you would be better off using a different one.) Thus 128.6.4.27 will send back a redirect specifying that packets for 128.6.7.23 should be sent via 128.6.4.30. Your TCP/IP software will add a routing entry

128.6.7.23	128.6.4.30	UDHG	pe0

Any future datagrams for 128.6.7.23 will be sent directly to the appropriate gateway.

This strategy would be a complete solution, if it weren't for three problems:

- It requires each computer to have the address of one gateway "hardwired" into its startup files, as the initial default.

- If a gateway goes down, routing table entries using it may not be removed.

- If your network uses subnets, and your TCP/IP implementation does not handle them, this strategy will not work.

How serious the first problem is depends upon your situation. For all networks, there is no problem modifying startup files whenever something changes. But some organizations can find it very painful. If network topology changes, and a gateway is removed, any systems that have that gateway as their default must be adjusted. This is particularly serious if the people who maintain the network are not the same as those maintaining the individual systems. One simple appoach is to make sure that the default address never changes. For example, you might adopt the convention that address 1 on each subnet is the default gateway for that subnet. For example, on subnet 128.6.7, the default gateway would always be 128.6.7.1. If that gateway is ever removed, some other gateway is given that address. (There must always be at least one gateway left to give it to. If there isn't, you are completely cut off anyway.)

The biggest problem with the description given so far is that it tells you how to add routes but not how to get rid of them. What happens if a gateway goes down? You want traffic to be redirected back to a gateway that is up. Unfortunately, a gateway that has crashed is not going to issue Redirects. One solution is to choose very reliable gateways. If they crash very seldom, this may not be a problem. Note that Redirects can be used to handle some kinds of network failure. If a line goes down, your current route may no longer be a good one. As long as the gateway to which you are talking is still up and talking to you, it can simply issue a Redirect to the gateway that is now the best one. However you still need a way to detect failure of one of the gateways that you are talking to directly.

The best approach for handling failed gateways is for your TCP/IP implementation to detect routes that have failed. TCP maintains various timers that allow the software to detect when a connection has broken. When this happens, one good approach is to mark the route down, and go back to the default gateway. A similar approach can also be used to handle failures in the default gateway. If you have mark two gateways as default, then the software should be capable of switching when connections using one of them start failing. Unfortunately, some common TCP/IP implementations do not mark routes as down and change to new ones. (In particular Berkeley 4.2 Unix does not.) However Berkeley 4.3 Unix does do this, and as other vendors begin to base products on 4.3 rather than 4.2, this ability is expected to be more common.

4.4 Other ways for hosts to find routes

As long as your TCP/IP implementations handle failing connections properly, establishing one or more default routes in the configuration file is likely to be the simplest way to handle routing. However there are two other routing approaches that are worth considering for special situations:

- spying on the routing protocol
- using proxy ARP

4.4.1 Spying on Routing

Gateways generally have a special protocol that they use among themselves. Note that redirects cannot be used by gateways. Redirects are simply ways for gateways to tell "dumb" hosts to use a different gateway. The gateways themselves must have a complete picture of the network, and a way to compute the optimal route to each subnet. Generally they maintain this picture by exchanging information among themselves. There are several different routing protocols in use for this purpose. One way for a computer to keep track of gateways is for it to listen to the gateways' messages. There is software available for this purpose for most of the common routing protocols. When you run this software, it maintains a complete picture of the network, just as the gateways do. The software is generally designed to maintain your computer's routing tables dynamically, so that datagrams are always sent to the proper gateway. In effect, the routing software issues the equivalent of the Unix "route add" and "route delete" commands as the network topology changes. Generally this results in a complete routing table, rather than one that depends upon default routes. (This assumes that the gateways themselves maintain a complete table. Sometimes gateways keep track of your campus network completely, but use a default route for all off-campus networks, etc.)

Running routing software on each host does in some sense "solve" the routing problem. However there are several reasons why this is not normally recommended except as a last resort. The most serious problem is that this reintroduces configuration options that must be kept up to date on each host. Any computer that wants to participate in the protocol among the gateways will need to configure its software compatibly with the gateways. Modern gateways often have configuration options that are complex compared with those of an individual host. It is undesirable to spread these to every host.

There is a somewhat more specialized problem that applies only to diskless computers. By its very nature, a diskless computer depends upon the network and file servers to load programs and to do swapping. It is dangerous for diskless computers to run any software that listens to network broadcasts. Routing software generally depends upon broadcasts. For example, each gateway on the network might broadcast its routing tables every 30 seconds. The problem with diskless nodes is that the software to listen to these broadcasts must be loaded over the network. On a busy computer, programs that are not used for a few seconds will be swapped or paged out. When they are activated again, they must be swapped or paged in. Whenever a broadcast is sent, every computer on the network needs to activate the routing software in order to process the broadcast. This means that many diskless computers will be doing swapping or paging at the same time. This is likely to cause a temporary overload of the network. Thus it is very unwise for diskless machines to run any software that requires them to listen to broadcasts.

4.4.2 Proxy ARP

Proxy ARP is an alternative technique for letting gateways make all the routing decisions. It is applicable to any broadcast network that uses ARP or a similar technique for mapping Internet addresses into network-specific addresses such as Ethernet addresses. This presentation will assume Ethernet. Other network types can be accomodated if you replace "Ethernet address" with the appropriate network-specific address, and ARP with the protocol used for address mapping by that network type.

In many ways proxy ARP it is similar to using a default route and redirects, however it uses a different mechanism to communicate routes to the host. With redirects, a full routing table is used. At any given moment, the host knows what gateways it is routing datagrams to. With proxy ARP, you dispense with explicit routing tables, and do everything at the level of Ethernet addresses. Proxy ARP can be used for all destinations, only for destinations within your network, or in various combinations. It will be simplest to explain it as used for all addresses. To do this, you instruct the host to pretend that every computer in the world is attached directly to your local Ethernet. On Unix, this would be done using a command

```
route add default 128.6.4.2 0
```

where 128.6.4.2 is assumed to be the Internet address of your host. As explained above, the metric of 0 causes everything that matches this route to be sent directly on the local Ethernet.

When a datagram is to be sent to a local Ethernet destination, your computer needs to know the Ethernet address of the destination. In order to find that, it uses something generally called the ARP table. This is simply a mapping from Internet address to Ethernet address. Here's a typical ARP table. (On our system, it is displayed using the command "arp -a".)

```
FOKKER.RUTGERS.EDU (128.6.5.16) at 8:0:20:0:8:22 temporary
CROSBY.RUTGERS.EDU (128.6.5.48) at 2:60:8c:49:50:63 temporary
CAIP.RUTGERS.EDU (128.6.4.16) at 8:0:8b:0:1:6f temporary
DUDE.RUTGERS.EDU (128.6.20.16) at 2:7:1:0:eb:cd temporary
W20NS.MIT.EDU (18.70.0.160) at 2:7:1:0:eb:cd temporary
OBERON.USC.EDU (128.125.1.1) at 2:7:1:2:18:ee temporary
gatech.edu (128.61.1.1) at 2:7:1:0:eb:cd temporary
DARTAGNAN.RUTGERS.EDU (128.6.5.65) at 8:0:20:0:15:a9 temporary
```

Note that it is simply a list of Internet addresses and the corresponding Ethernet address. The "temporary" indicates that the entry was added dynamically using ARP, rather than being put into the table manually.

If there is an entry for the address in the ARP table, the datagram is simply put on the Ethernet with the corresponding Ethernet address. If not, an "ARP request" is broadcast, asking for the destination host to identify itself. This request is in effect a question "will the host with Internet address 128.6.4.194 please tell me what your Ethernet address is?". When a response comes back, it is added to the ARP table, and future datagrams for that destination can be sent without delay.

This mechanism was originally designed only for use with hosts attached directly to a single Ethernet. If you need to talk to a host on a different Ethernet, it was assumed that your routing table would direct you to a gateway. The gateway would of course have one interface on your Ethernet. Your computer would then end up looking up the address of that gateway using ARP. It would generally be useless to expect ARP to work directly with a computer on a distant network. Since it isn't on the same Ethernet, there's no Ethernet address you can use to send datagrams to it. And when you send an ARP request for it, there's nobody to answer the request.

Proxy ARP is based on the concept that the gateways will act as proxies for distant hosts. Suppose you have a host on network 128.6.5, with address 128.6.5.2. (computer A in diagram below) It wants to send a datagram to host 128.6.4.194, which is on a different Ethernet (subnet 128.6.4). (computer C in diagram below) There is a gateway connecting the two subnets, with address 128.6.5.1 (gateway R):

```
        network 1                          network 2
         128.6.5                            128.6.4
 ---------------------------------  ------------------
    |                 |         |     |       |     |
    |                 |         |     |       |     |
 ___|_____       _____|____   ___|_____|__   ___|_____|____
 128.6.5.2       128.6.5.3     128.6.5.1     128.6.4.194
                               128.6.4.1

 ____________    ____________  ____________  ______________
 computer A      computer B    gateway R     computer C
```

Now suppose computer A sends an ARP request for computer C. C isn't able to answer for itself. It's on a different network, and never even sees the ARP request. However gateway R can act on its behalf. In effect, your computer asks "will the host with Internet address 128.6.4.194 please tell me what your Ethernet address is?", and the gateway says "here I am, 128.6.4.194 is 2:7:1:0:eb:cd", where 2:7:1:0:eb:cd is actually the Ethernet address of the gateway. This bit of illusion works just fine. Your host now thinks that 128.6.4.194 is attached to the local Ethernet with address 2:7:1:0:eb:cd. Of course it isn't. But it works anyway. Whenever there's a datagram to be sent to 128.6.4.194, your host sends it to the specified Ethernet address. Since that's the address of a gateway R, the gateway gets the packet. It then forwards it to the destination.

Note that the net effect is exactly the same as having an entry in the routing table saying to route destination 128.6.4.194 to gateway 8.6.5.1:

```
128.6.4.194          128.6.5.1          UGH          pe0
```

except that instead of having the routing done at the level of the routing table, it is done at the level of the ARP table.

Generally it's better to use the routing table. That's what it's there for. However here are some cases where proxy ARP makes sense:

- when you have a host that does not implement subnets
- when you have a host that does not respond properly to redirects
- when you do not want to have to choose a specific default gateway
- when your software is unable to recover from a failed route

The technique was first designed to handle hosts that do not support subnets. Suppose that you have a subnetted network. For example, you have chosen to break network 128.6 into subnets, so that 128.6.4 and 128.6.5 are separate. Suppose you have a computer that does not understand subnets. It will assume that all of 128.6 is a single network. Thus it will be difficult to establish routing table entries to handle the configuration above. You can't tell it about the gateway explicitly using "route add 128.6.4.0 128.6.5.1 1" Since it thinks all of 128.6 is a single network, it can't understand that you

are trying to tell it where to send one subnet. It will instead interpret this command as an attempt to set up a host route to a host who address is 128.6.4.0. The only thing that would work would be to establish explicit host routes for every individual host on every other subnet. You can't depend upon default gateways and redirects in this situation either. Suppose you said "route add default 128.6.5.1 1". This would establish the gateway 128.6.5.1 as a default. However the system wouldn't use it to send packets to other subnets. Suppose the host is 128.6.5.2, and wants to send a datagram to 128.6.4.194. Since the destination is part of 128.6, your computer considers it to be on the same network as itself, and doesn't bother to look for a gateway.

Proxy ARP solves this problem by making the world look the way the defective implementation expects it to look. Since the host thinks all other subnets are part of its own network, it will simply issue ARP requests for them. It expects to get back an Ethernet address that can be used to establish direct communications. If the gateway is practicing proxy ARP, it will respond with the gateway's Ethernet address. Thus datagrams are sent to the gateway, and everything works.

As you can see, no specific configuration is need to use proxy ARP with a host that doesn't understand subnets. All you need is for your gateways to implement proxy ARP. In order to use it for other purposes, you must explicitly set up the routing table to cause ARP to be used. By default, TCP/IP implementations will expect to find a gateway for any destination that is on a different network. In order to make them issue ARP's, you must explicitly install a route with metric 0, as in the example "route add default 128.6.5.2 0".

It is obvious that proxy ARP is reasonable in situations where you have hosts that don't understand subnets. Some comments may be needed on the other situations. Generally TCP/IP implementations do handle ICMP redirects properly. Thus it is normally practical to set up a default route to some gateway, and depend upon the gateway to issue redirects for destinations that should use a different gateway. However in case you ever run into an implementation that does not obey redirects, or cannot be configured to have a default gateway, you may be able to make things work by depending upon proxy ARP. Of course this requires that you be able to configure the host to issue ARP's for all destinations. You will need to read the documentation carefully to see exactly what routing features your implementation has.

Sometimes you may choose to depend upon proxy ARP for convenience. The problem with routing tables is that you have to configure them. The simplest configuration is simply to establish a default route, but even there you have to supply some equivalent to the Unix command "route add default ...". Should you change the addresses of your gateways, you have to modify this command on all of your hosts, so that they point to the new default gateway. If you set up a default route that depends upon proxy ARP (i.e. has metric 0), you won't have to change your configuration files when gateways change. With proxy ARP, no gateway addresses are given explicitly. Any gateway can

respond to the ARP request, no matter what its address.

In order to save you from having to do configuration, some TCP/IP implementations default to using ARP when they have no other route. The most flexible implementations allow you to mix strategies. That is, if you have specified a route for a particular network, or a default route, they will use that route. But if there is no route for a destination, they will treat it as local, and issue an ARP request. As long as your gateways support proxy ARP, this allows such hosts to reach any destination without any need for routing tables.

Finally, you may choose to use proxy ARP because it provides better recovery from failure. This choice is very much dependent upon your implementation. The next section will discuss the tradeoffs in more detail.

In situations where there are several gateways attached to your network, you may wonder how proxy ARP allows you to choose the best one. As described above, your computer simply sends a broadcast asking for the Ethernet address for a destination. We assumed that the gateways would be set up to respond to this broadcast. If there is more than one gateway, this requires coordination among them. Ideally, the gateways will have a complete picture of the network topology. Thus they are able to determine the best route from your host to any destination. If the gateway coordinate among themselves, it should be possible for the best gateway to respond to your ARP request. In practice, it may not always be possible for this to happen. It is fairly easy to design algorithms to prevent very bad routes. For example, consider the following situation:

```
        1              2              3
------- A ---------- B ----------
```

1, 2, and 3 are networks. A and B are gateways, connecting network 2 to 1 or 3. If a host on network 2 wants to talk to a host on network 1, it is fairly easy for gateway A to decide to answer, and for gateway B to decide not to. Here's how: if gateway B accepted a datagram for network 1, it would have to forward it to gateway A for delivery. This would mean that it would take a packet from network 2 and send it right back out on network 2. It is very easy to test for routes that involve this sort of circularity. It is much harder to deal with a situation such as the following:

```
            1
---------------
  A         B
  |         | 4
  |         |
3 |         C
  |         |
  |         | 5
  D         E
---------------
            2
```

Suppose a computer on network 1 wants to send a datagram to one on network 2. The route via A and D is probably better, because it goes through only one intermediate network (3). It is also possible to go via B, C, and E, but that path is probably slightly slower. Now suppose the computer on network 1 sends an ARP request for a destination on 2. It is likely that A and B will both respond to that request. B is not quite as good a route as A. However it is not so bad as the case above. B won't have to send the datagram right back out onto network 1. It is unable to determine there is a better alternative route without doing a significant amount of global analysis on the network. This may not be practical in the amount of time available to process an ARP request.

4.4.3 Moving to New Routes After Failures

In principle, TCP/IP routing is capable of handling line failures and gateway crashes. There are various mechanisms to adjust routing tables and ARP tables to keep them up to date. Unfortunately, many major implementations of TCP/IP have not implemented all of these mechanisms. The net result is that you have to look carefully at the documentation for your implementation, and consider what kinds of failures are most likely. You then have to choose a strategy that will work best for your site. The basic choices for finding routes have all been listed above: spying on the gateways' routing protocol, setting up a default route and depending upon redirects, and using proxy ARP. These methods all have their own limitations in dealing with a changing network.

Spying on the gateways' routing protocol is theoretically the cleanest solution. Assuming that the gateways use good routing technology, the tables that they broadcast contain enough information to maintain optimal routes to all destinations. Should something in the network change .(a line or a gateway goes down), this information will be reflected in the tables, and the routing software will be able to update the hosts' routing tables appropriately. The disadvantages are entirely practical. However in some situations the robustness of this approach may outweight the disadvantages. To summarize the discussion above, the disadvantages are:

- If the gateways are using sophisticated routing protocols, configuration may be fairly complex. Thus you will be faced with setting up and maintaining configuration files on every host.

- Some gateways use proprietary routing protocols. In this case, you may not be able to find software for your hosts that understands them.

- If your hosts are diskless, there can be very serious performance problems associated with listening to routing broadcasts.

Some gateways may be able to convert from their internal routing protocol to a simpler one for use by your hosts. This could largely

bypass the first two disadvantages. Currently there is no known way to get around the third one.

The problems with default routes/redirects and with proxy ARP are similar: they both have trouble dealing with situations where their table entries no longer apply. The only real difference is that different tables are involved. Suppose a gateway goes down. If any of your current routes are using that gateway, you may be in trouble. If you are depending upon the routing table, the major mechanism for adjusting routes is the redirect. This works fine in two situations:

- where the default gateway is not the best route. The default gateway can direct you to a better gateway

- where a distant line or gateway fails. If this changes the best route, the current gateway can redirect you to the gateway that is now best

The case it does not protect you against is where the gateway that you are currently sending your datagrams to crashes. Since it is down, it is unable to redirect you to another gateway. In many cases, you are also unprotected if your default gateway goes down, since there routing starts by sending to the default gateway.

The situation with proxy ARP is similar. If the gateways coordinate themselves properly, the right one will respond initially. If something elsewhere in the network changes, the gateway you are currently issuing can issue a redirect to a new gateway that is better. (It is usually possible to use redirects to override routes established by proxy ARP.) Again, the case you are not protected against is where the gateway you are currently using crashes. There is no equivalent to failure of a default gateway, since any gateway can respond to the ARP request.

So the big problem is that failure of a gateway you are using is hard to recover from. It's hard because the main mechanism for changing routes is the redirect, and a gateway that is down can't issue redirects. Ideally, this problem should be handled by your TCP/IP implementation, using timeouts. If a computer stops getting response, it should cancel the existing route, and try to establish a new one. Where you are using a default route, this means that the TCP/IP implementation must be able to declare a route as down based on a timeout. If you have been redirected to a non-default gateway, and that route is declared down, traffic will return to the default. The default gateway can then begin handling the traffic, or redirect it to a different gateway. To handle failure of a default gateway, it should be possible to have more than one default. If one is declared down, another will be used. Together, these mechanisms should take care of any failure.

Similar mechanisms can be used by systems that depend upon proxy ARP. If a connection is timing out, the ARP table entry that it uses should be cleared. This will cause a new ARP request, which can be handled by a gateway that is still up. A simpler mechanism would simply be to time out all ARP entries after some period. Since making a new ARP

request has a very low overhead, there's no problem with removing an ARP entry even if it is still good. The next time a datagram is to be sent, a new request will be made. The response is normally fast enough that users will not even notice the delay.

Unfortunately, many common implementations do not use these strategies. In Berkeley 4.2, there is no automatic way of getting rid of any kind of entry, either routing or ARP. They do not invalidate routes on timeout nor ARP entries. ARP entries last forever. If gateway crashes are a significant problem, there may be no choice but to run software that listens to the routing protocol. In Berkeley 4.3, routing entries are removed when TCP connections are failing. ARP entries are still not removed. This makes the default route strategy more attractive for 4.3 than proxy ARP. Having more than one default route may also allow for recovery from failure of a default gateway. Note however that 4.3 only handles timeout for connections using TCP. If a route is being used only by services based on UDP, it will not recover from gateway failure. While the "traditional" TCP/IP services use TCP, network file systems generally do not. Thus 4.3-based systems still may not always be able to recover from failure.

In general, you should examine your implementation in detail to determine what sort of error recovery strategy it uses. We hope that the discussion in this section will then help you choose the best way of dealing with routing.

There is one more strategy that some older implementations use. It is strongly discouraged, but we mention it here so you can recognize it if you see it. Some implementations detect gateway failure by taking active measure to see what gateways are up. The best version of this is based on a list of all gateways that are currently in use. (This can be determined from the routing table.) Every minute or so, an echo request datagram is sent to each such gateway. If a gateway stops responding to echo requests, it is declared down, and all routes using it revert to the default. With such an implementation, you normally supply more than one default gateway. If the current default stops responding, an alternate is chosen. In some cases, it is not even necessary to choose an explicit default gateway. The software will randomly choose any gateway that is responding. This implementation is very flexible and recovers well from failures. However a large network full of such implementations will waste a lot of bandwidth on the echo datagrams that are used to test whether gateways are up. This is the reason that this strategy is discouraged.

5. Bridges and Gateways

This section will deal in more detail with the technology used to construct larger networks. It will focus particularly on how to connect together multiple Ethernets, token rings, etc. These days most networks are hierarchical. Individual hosts attach to local-area

networks such as Ethernet or token ring. Then those local networks are connected via some combination of backbone networks and point to point links. A university might have a network that looks in part like this:

```
 ____________________________________________
|   net 1        net 2     net 3  |        net 4            net 5
| ---------X---------X-------- |      --------          --------
|                          |    |        |                  |
|  Building A              |    |        |                  |
|                ----------X----------------X-----------------X
|                          |    | campus backbone network   :
|                          |    |                           :
|____________________________|                        serial :
                                                       line :
                                                    -------X-----
                                                        net  6
```

Nets 1, 2 and 3 are in one building. Nets 4 and 5 are in different buildings on the same campus. Net 6 is in a somewhat more distant location. The diagram above shows nets 1, 2, and 3 being connected directly, with switches that handle the connections being labelled as "X". Building A is connected to the other buildings on the same campus by a backbone network. Note that traffic from net 1 to net 5 takes the following path:

- from 1 to 2 via the direct connection between those networks
- from 2 to 3 via another direct connection
- from 3 to the backbone network
- across the backbone network from building A to the building in which net 5 is housed
- from the backbone network to net 5

Traffic for net 6 would additionally pass over a serial line. With the setup as shown, the same switch is being used to connect the backbone network to net 5 and to the serial line. Thus traffic from net 5 to net 6 would not need to go through the backbone, since there is a direct connection from net 5 to the serial line.

This section is largely about what goes in those "X"'s.

5.1 Alternative Designs

Note that there are alternatives to the sort of design shown above. One is to use point to point lines or switched lines directly to each host. Another is to use a single-level of network technology that is capable of handling both local and long-haul networking.

5.1.1 A mesh of point to point lines

Rather than connecting hosts to a local network such as Ethernet, and then interconnecting the Ethernets, it is possible to connect long-haul serial lines directly to the individual computers. If your network consists primarily of individual computers at distant locations, this might make sense. Here would be a small design of that type.

```
computer 1                  computer 2                  computer 3
    |                           |                           |
    |                           |                           |
    |                           |                           |
computer 4 -------------- computer 5 ----------- computer 6
```

In the design shown earlier, the task of routing datagrams around the network is handled by special-purpose switching units shown as "X"'s. If you run lines directly between pairs of hosts, your hosts will be doing this sort of routing and switching, as well as their normal computing. Unless you run lines directly between every pair of computers, some systems will end up handling traffic for others. For example, in this design, traffic from 1 to 3 will go through 4, 5 and 6. This is certainly possible, since most TCP/IP implementations are capable of forwarding datagrams. If your network is of this type, you should think of your hosts as also acting as gateways. Much of the discussion below on configuring gateways will apply to the routing software that you run on your hosts. This sort of configuration is not as common as it used to be, for two reasons:

- Most large networks have more than one computer per location. In this case it is less expensive to set up a local network at each location than to run point to point lines to each computer.

- Special-purpose switching units have become less expensive. It often makes sense to offload the routing and communications tasks to a switch rather than handling it on the hosts.

It is of course possible to have a network that mixes the two kinds of techology. In this case, locations with more equipment would be handled by a hierarchical system, with local-area networks connected by switches. Remote locations with a single computer would be handled by point to point lines going directly to those computers. In this case the routing software used on the remote computers would have to be compatible with that used by the switches, or there would need to be a gateway between the two parts of the network.

Design decisions of this type are typically made after an assessment of the level of network traffic, the complexity of the network, the quality of routing software available for the hosts, and the ability of the hosts to handle extra network traffic.

5.1.2 Circuit switching technology

Another alternative to the hierarchical LAN/backbone approach is to use circuit switches connected to each individual computer. This is really a variant of the point to point line technique, where the circuit switch allows each system to have what amounts to a direct line to every other system. This technology is not widely used within the TCP/IP community, largely because the TCP/IP protocols assume that the lowest level handles isolated datagrams. When a continuous connection is needed, higher network layers maintain it using datagrams. This datagram-oriented technology does not match a circuit-oriented environment very closely. In order to use circuit switching technology, the IP software must be modified to be able to build and tear down virtual circuits as appropriate. When there is a datagram for a given destination, a virtual circuit must be opened to it. The virtual circuit would be closed when there has been no traffic to that destination for some time. The major use of this technology is for the DDN (Defense Data Network). The primary interface to the DDN is based on X.25. This network appears to the outside as a distributed X.25 network. TCP/IP software intended for use with the DDN must do precisely the virtual circuit management just described. Similar techniques could be used with other circuit-switching technologies, e.g. ATT's DataKit, although there is almost no software currently available to support this.

5.1.3 Single-level networks

In some cases new developments in wide-area networks can eliminate the need for hierarchical networks. Early hierarchical networks were set up because the only convenient network technology was Ethernet or other LAN's, and those could not span distances large enough to cover an entire campus. Thus it was necessary to use serial lines to connect LAN's in various locations. It is now possible to find network technology whose characteristics are similar to Ethernet, but where a single network can span a campus. Thus it is possible to think of using a single large network, with no hierarchical structure.

The primary limitations of a large single-level network are performance and reliability considerations. If a single network is used for the entire campus, it is very easy to overload it. Hierarchical networks can handle a larger traffic volume than single-level networks if traffic patterns have a reasonable amount of locality. That is, in many applications, traffic within an individual department tends to be greater than traffic among departments.

Let's look at a concrete example. Suppose there are 10 departments, each of which generate 1 Mbit/sec of traffic. Suppose futher than 90% of that traffic is to other systems within the department, and only 10% is to other departments. If each department has its own network, that network only needs to handle 1 Mbit/sec. The backbone network connecting the department also only needs 1 Mbit/sec capacity, since

it is handling 10% of 1 Mbit from each department. In order to handle this situation with a single wide-area network, that network would have to be able to handle the simultaneous load from all 10 departments, which would be 10 Mbit/sec.

The second limitation on single-level networks is reliability, maintainability and security. Wide-area networks are more difficult to diagnose and maintain than local-area networks, because problems can be introduced from any building to which the network is connected. They also make traffic visible in all locations. For these reasons, it is often sensible to handle local traffic locally, and use the wide-area network only for traffic that actually must go between buildings. However if you have a situation where each location has only one or two computers, it may not make sense to set up a local network at each location, and a single-level network may make sense.

5.1.4 Mixed designs

In practice, few large networks have the luxury of adopting a theoretically pure design.

It is very unlikely that any large network will be able to avoid using a hierarchical design. Suppose we set out to use a single-level network. Even if most buildings have only one or two computers, there will be some location where there are enough that a local-area network is justified. The result is a mixture of a single-level network and a hierachical network. Most buildings have their computers connected directly to the wide-area network, as with a single-level network. However in one building there is a local-area network which uses the wide-area network as a backbone, connecting to it via a switching unit.

On the other side of the story, even network designers with a strong commitment to hierarchical networks are likely to find some parts of the network where it simply doesn't make economic sense to install a local-area network. So a host is put directly onto the backbone network, or tied directly to a serial line.

However you should think carefully before making ad hoc departures from your design philosophy in order to save a few dollars. In the long run, network maintainability is going to depend upon your ability to make sense of what is going on in the network. The more consistent your technology is, the more likely you are to be able to maintain the network.

5.2 An introduction to alternative switching technologies

This section will discuss the characteristics of various technologies used to switch datagrams between networks. In effect, we are trying to fill in some details about the black boxes assumed in previous sections. There are three basic types of switches, generally referred to as repeaters, bridges, and gateways, or alternatively as level 1, 2 and 3 switches (based on the level of the ISO model at which they operate). Note however that there are systems that combine features of more than one of these, particularly bridges and gateways.

The most important dimensions on which switches vary are isolation, performance, routing and network management facilities. These will be discussed below.

The most serious difference is between repeaters and the other two types of switch. Until recently, gateways provided very different services from bridges. However these two technologies are now coming closer together. Gateways are beginning to adopt the special-purpose hardware that has characterized bridges in the past. Bridges are beginning to adopt more sophisticated routing, isolation features, and network management, which have characterized gateways in the past. There are also systems that can function as both bridge and gateway. This means that at the moment, the crucial decision may not be to decide whether to use a bridge or a gateway, but to decide what features you want in a switch and how it fits into your overall network design.

5.2.1 Repeaters

A repeater is a piece of equipment that connects two networks that use the same technology. It receives every data packet on each network, and retransmits it onto the other network. The net result is that the two networks have exactly the same set of packets on them. For Ethernet or IEEE 802.3 networks there are actually two different kinds of repeater. (Other network technologies may not need to make this distinction.)

A simple repeater operates at a very low level indeed. Its primary purpose is to get around limitations in cable length caused by signal loss or timing dispersion. It allows you to construct somewhat larger networks than you would otherwise be able to construct. It can be thought of as simply a two-way amplifier. It passes on individual bits in the signal, without doing any processing at the packet level. It even passes on collisions. That is, if a collision is generated on one of the networks connected to it, the repeater generates a collision on the other network. There is a limit to the number of repeaters that you can use in a network. The basic Ethernet design requires that signals must be able to get from one end of the network to the other within a specified amount of time. This determines a maximum allowable length. Putting repeaters in the path does not get

around this limit. (Indeed each repeater adds some delay, so in some ways a repeater makes things worse.) Thus the Ethernet configuration rules limit the number of repeaters that can be in any path.

A "buffered repeater" operates at the level of whole data packets. Rather than passing on signals a bit at a time, it receives an entire packet from one network into an internal buffer and then retransmits it onto the other network. It does not pass on collisions. Because such low-level features as collisions are not repeated, the two networks continue to be separate as far as the Ethernet specifications are concerned. Thus there are no restrictions on the number of buffered repeaters that can be used. Indeed there is no requirement that both of the networks be of the same type. However the two networks must be sufficiently similar that they have the same packet format. Generally this means that buffered repeaters can be used between two networks of the IEEE 802.x family (assuming that they have chosen the same address length), or two networks of some other related family. A pair of buffered repeaters can be used to connect two networks via a serial line.

Buffered repeaters share with simple repeaters the most basic feature: they repeat every data packet that they receive from one network onto the other. Thus the two networks end up with exactly the same set of packets on them.

5.2.2 Bridges and gateways

A bridge differs from a buffered repeater primarily in the fact that it exercizes some selectivity as to what packets it forwards between networks. Generally the goal is to increase the capacity of the system by keeping local traffic confined to the network on which it originates. Only traffic intended for the other network (or some other network accessed through it) goes through the bridge. So far this description would also apply to a gateway. Bridges and gateways differ in the way they determine what packets to forward. A bridge uses only the ISO level 2 address. In the case of Ethernet or IEEE 802.x networks, this is the 6-byte Ethernet or MAC-level address. (The term MAC-level address is more general. However for the sake of concreteness, examples in this section will assume that Ethernet is being used. You may generally replace the term "Ethernet address" with the equivalent MAC-level address for other similar technologies.) A bridge does not examine the packet itself, so it does not use the IP address or its equivalent for routing decisions. In contrast, a gateway bases its decisions on the IP address, or its equivalent for other protocols.

There are several reasons why it matters which kind of address is used for decisions. The most basic is that it affects the relationship between the switch and the upper layers of the protocol. If forwarding is done at the level of the MAC-level address (bridge), the switch will be invisible to the protocols. If it is done at the IP level, the switch will be visible. Let's give an example. Here are

two networks connected by a bridge:

```
          network 1                 network 2
           128.6.5                   128.6.4
 ----------------- ------------------------------
 |                 |       |                |
 |                 |       |                |
 |_________        |_______|_       _________|____
 128.6.5.2          bridge          128.6.4.3   128.6.4.4

 ________________   ________________ ________________ ________________
 computer A                          computer B      computer C
```

Note that the bridge does not have an IP address. As far as computers A, B, and C are concerned, there is a single Ethernet (or other network) to which they are all attached. This means that the routing tables must be set up so that computers on both networks treat both networks as local. When computer A opens a connection to computer B, it first broadcasts an ARP request asking for computer B's Ethernet address. The bridge must pass this broadcast from network 1 to network 2. (In general, bridges must pass all broadcasts.) Once the two computers know each other's Ethernet addresses, communications use the Ethernet address as the destination. At that point, the bridge can start exerting some selectivity. It will only pass packets whose Ethernet destination address is for a machine on the other network. Thus a packet from B to A will be passed from network 2 to 1, but a packet from B to C will be ignored.

In order to make this selection, the bridge needs to know which network each machine is on. Most modern bridges build up a table for each network, listing the Ethernet addresses of machines known to be on that network. They do this by watching all of the packets on both networks. When a packet first appears on network 1, it is reasonable to conclude that the Ethernet source address corresponds to a machine on network 1.

Note that a bridge must look at every packet on the Ethernet, for two different reasons. First, it may use the source address to learn which machines are on which network. Second, it must look at the destination address in order to decide whether it needs to forward the packet to the other network.

As mentioned above, generally bridges must pass broadcasts from one network to the other. Broadcasts are often used to locate a resource. The ARP request is a typical example of this. Since the bridge has no way of knowing what host is going to answer the broadcast, it must pass it on to the other network. Some newer bridges have user-selectable filters. With them, it is possible to block some broadcasts and allow others. You might allow ARP broadcasts (which are essential for IP to function), but confine less essential broadcasts to one network. For example, you might choose not to pass rwhod broadcasts, which some systems use to keep track of every user logged into every other system. You might decide that it is sufficient for rwhod to know about the systems on a single segment of the network.

Now let's take a look at two networks connected by a gateway

```
        network 1                                  network 2
        128.6.5                                    128.6.4
  -------------------                  ---------------------------------
        |                  |                 |                 |                 |
   ____|_____        ____|_____        ____|____        _____|____        _____|____
   128.6.5.2         128.6.5.1  128.6.4.1         128.6.4.3         128.6.4.4
   ___________       ___________________________   ___________       ___________
   computer A                gateway                computer B        computer C
```

Note that the gateway has IP addresses assigned to each interface. The computers' routing tables are set up to forward through appropriate address. For example, computer A has a routing entry saying that it should use the gateway 128.6.5.1 to get to subnet 128.6.4.

Because the computers know about the gateway, the gateway does not need to scan all the packets on the Ethernet. The computers will send packets to it when appropriate. For example, suppose computer A needs to send a message to computer B. Its routing table will tell it to use gateway 128.6.5.1. It will issue an ARP request for that address. The gateway will respond to the ARP request, just as any host would. From then on, packets destinated for B will be sent with the gateway's Ethernet address.

5.2.3 More about bridges

There are several advantages to using the Mac-level address, as a bridge does. First, every packet on an Ethernet or IEEE network has such an address. The address is in the same place for every packet, whether it is IP, DECnet, or some other protocol. Thus it is relatively fast to get the address from the packet. A gateway must decode the entire IP header, and if it is to support protocols other than IP, it must have software for each such protocol. This means that a bridge automatically supports every possible protocol, whereas a gateway requires specific provisions for each protocol it is to support.

However there are also disadvantages. The one that is intrinsic to the design of a bridge is

- A bridge must look at every packet on the network, not just those addressed to it. Thus it is possible to overload a bridge by putting it on a very busy network, even if very little traffic is actually going through the bridge.

However there are another set of disadvantages that are based on the way bridges are usually built. It is possible in principle to design bridges that do not have these disadvantages, but I don't know of any plans to do so. They all stem from the fact that bridges do not have

a complete routing table that describes the entire system of networks. They simply have a list of the Ethernet addresses that lie on each of its two networks. This means

- A bridge can handle only two network interfaces. At a central site, where many networks converge, this normally means that you set up a backbone network to which all the bridges connect, and then buy a separate bridge to connect each other network to that backbone. Gateways often have between 4 and 8 interfaces.

- Networks that use bridges cannot have loops in them. If there were a loop, some bridges would see traffic from the same Ethernet address coming from both directions, and would be unable to decide which table to put that address in. Note that any parallel paths to the same direction constitute a loop. This means that multiple paths cannot be used for purposes of splitting the load or providing redundancy.

There are some ways of getting around the problem of loops. Many bridges allow configurations with redundant connections, but turn off links until there are no loops left. Should a link fail, one of the disabled ones is then brought back into service. Thus redundant links can still buy you extra reliability. But they can't be used to provide extra capacity. It is also possible to build a bridge that will make use of parallel point to point lines, in the one special case where those lines go between a single pair of bridges. The bridges would treat the two lines as a single virtual line, and use them alternately in round-robin fashion.

The process of disabling redundant connections until there are no loops left is called a "spanning tree algorithm". This name comes from the fact that a tree is defined as a pattern of connections with no loops. Thus one wants to disable connections until the connections that are left form a tree that "spans" (includes) all of the networks in the system. In order to do this, all of the bridges in a network system must communicate among themselves. There is an IEEE proposal to standardize the protocol for doing this, and for constructing the spanning tree.

Note that there is a tendency for the resulting spanning tree to result in high network loads on certain parts of the system. The networks near the "top of the tree" handle all traffic between distant parts of the network. In a network that uses gateways, it would be possible to put in an extra link between parts of the network that have heavy traffic between them. However such extra links cannot be used by a set of bridges.

5.2.4 More about gateways

Gateways have their own advantages and disadvantages. In general a gateway is more complex to design and to administer than a bridge. A gateway must participate in all of the protocols that it is designed to forward. For example, an IP gateway must respond to ARP requests. The IP standards also require it to completely process the IP header, decrementing the time to live field and obeying any IP options.

Gateways are designed to handle more complex network topologies than bridges. As such, they have a different (and more complex) set of decisions to make. In general a bridge has only a binary decision to make: does it or does it not pass a given packet from one network to the other? However a gateway may have several network interfaces. Furthermore, when it forwards a packet, it must decide what host or gateway to send the packet to next. It is even possible for a gateway to decide to send a packet back onto the same network it came from. If a host is using the gateway as its default, it may send packets that really should go to some other gateway. In that case, the gateway will send the packet on to the right gateway, and send back an ICMP redirect to the host. Many gateways can also handle parallel paths. If there are several equally good paths to a destination, the gateway will alternate among them in round-robin fashion.

In order to handle these decisions, a gateway will typically have a routing table that looks very much like a host's. As with host routing tables, a gateway's table contains an entry for each possible network number. For each network, there is either an entry saying that that network is connected directly to the gateway, or there is an entry saying that traffic for that network should be forwarded through some other gateway or gateways. We will describe the "routing protocols" used to build up this information later, in the discussion on how to configure a gateway.

5.3 Comparing the switching technologies

Repeaters, buffered repeaters, bridges, and gateways form a spectrum. Those devices near the beginning of the list are best for smaller networks. They are less expensive, and easier to set up, but less general. Those near the end of the list are suitable for building more complex networks. Many networks will contain a mixture of switch types, with repeaters being used to connect a few nearby network segments, bridges used for slightly larger areas (particularly those with low traffic levels), and gateways used for long-distance links.

Note that this document so far has assumed that only gateways are being used. The section on setting up a host described how to set up a routing table listing the gateways to use to get to various networks. Repeaters and bridges are invisible to IP. So as far as previous sections are concerned, networks connected by them are to be considered a single network. Section 3.3.1 describes how to configure

a host in the case where several subnets are carried on a single physical network. The same configuration should be used when several subnets are connected by repeaters or bridges.

As mentioned above, the most important dimensions on which switches vary are isolation, performance, routing, network management, and performing auxilliary support services.

5.3.1 Isolation

Generally people use switches to connect networks to each other. So they are normally thinking of gaining connectivity, not providing isolation. However isolation is worth thinking about. If you connect two networks and provide no isolation at all, then any network problems on other networks suddenly appear on yours as well. Also, the two networks together may have enough traffic to overwhelm your network. Thus it is well to think of choosing an appropriate level of protection.

Isolation comes in two kinds: isolation against malfunctions and traffic isolation. In order to discuss isolation of malfunctions, we have to have a taxonomy of malfunctions. Here are the major classes of malfunctions, and which switches can isolate them:

- Electrical faults, e.g. a short in the cable or some sort of fault that distorts the signal. All types of switch will confine this to one side of the switch: repeater, buffered repeater, bridge, gateway. These are worth protecting against, although their frequency depends upon how often your cables are changed or disturbed. It is rare for this sort of fault to occur without some disturbance of the cable.

- Transceiver and controller problems that general signals that are valid electrically but nevertheless incorrect (e.g. a continuous, infinitely long packet, spurious collisions, never dropping carrier). All except the simple repeater will confine this: buffered repeater, bridge, gateway. (Such problems are not very common.)

- Software malfunctions that lead to excessive traffic between particular hosts (i.e. not broadcasts). Bridges and gateways will isolate these. (This type of failure is fairly rare. Most software and protocol problems generate broadcasts.)

- Software malfunctions that lead to excessive broadcast traffic. Gateways will isolate these. Generally bridges will not, because they must pass broadcasts. Bridges with user-settable filtering can protect against some broadcast malfunctions. However in general bridges must pass ARP, and most broadcast malfunctions involve ARP. This problem is not severe on single-vendor networks where software is under careful control. However research sites generally see problems of this sort regularly.

Traffic isolation is provided by bridges and gateways. The most basic decision is how many computers can be put onto a network without overloading its capacity. This requires knowledge of the capacity of the network, but also how the hosts will use it. For example, an Ethernet may support hundreds of systems if all the network is used for is remote logins and an occasional file transfer. However if the computers are diskless, and use the network for swapping, an Ethernet will support between 10 and 40, depending upon their speeds and I/O rates.

When you have to put more computers onto a network than it can handle, you split it into several networks and put some sort of switch between them. If you do the split correctly, most of the traffic will be between machines on the same piece. This means putting clients on the same network as their servers, putting terminal servers on the same network as the hosts that they access most commonly, etc.

Bridges and gateways generally provide similar degrees of traffic isolation. In both cases, only traffic bound for hosts on the other side of the switch is passed. However see the discussion on routing.

5.3.2 Performance

This is becoming less of an issue as time goes on, since the technology is improving. Generally repeaters can handle the full bandwidth of the network. (By their very nature, a simple repeater must be able to do so.) Bridges and gateways often have performance limitations of various sorts. Bridges have two numbers of interest: packet scanning rate and throughput. As explained above, a bridge must look at every packet on the network, even ones that it does not forward. The number of packets per second that it can scan in this way is the packet scanning rate. Throughput applies to both bridges and gateways. This is the rate at which they can forward traffic. Generally this depends upon packet size. Normally the number of packets per second that a unit can handle will be greater for short packets than long ones. Early models of bridge varied from a few hundred packets per second to around 7000. The higher speeds are for equipment that uses special-purpose hardware to speed up the process of scanning packets. First-generation gateways varied from a few hundred packets per second to 1000 or more. However second-generation gateways are now available, using special-purpose hardware of the same sophistication as that used by bridges. They can handle on the order of 10000 packets per second. Thus at the moment high-performance bridges and gateways can switch most of the bandwidth of an Ethernet. This means that performance should no longer be a basis for choosing between types of switch. However within a given type of switch, there are still specific models with higher or lower capacity.

Unfortunately there is no single number on which you can base performance estimates. The figure most commonly quoted is packets per second. Be aware that most vendors count a packet only once as it goes through a gateway, but that one prominent vendor counts packets

twice. Thus their switching rates must be deflated by a factor of 2. Also, when comparing numbers make sure that they are for packets of the same size. A simple performance model is

processing time = switching time + packet size * time per byte

That is, the time to switch a packet is normally a constant switching time, representing interrupt latency, header processing, routing table lookup, etc., plus a component proportional to packet size, representing the time needed to do any packet copying. One reasonable approach to reporting performance is to give packets per second for minimum and maximum size packets. Another is to report limiting switching speed in packets per second and throughput in bytes per second, i.e. the two terms of the equation above.

5.3.3 Routing

Routing refers to the technology used to decide where to send a packet next. Of course for a repeater this is not an issue, since repeaters forward every packet.

Bridges are almost always constructed with exactly two interfaces. Thus routing turns into two decisions: (1) whether the bridge should function at all, and (2) whether it should forward any particular packet. The second decision is usually based on a table of MAC-level addresses. As described above, this is built up by scanning traffic on both sides of the bridge. The goal is to forward those packets whose destination is on the other side of the bridge. This algorithm requires that the network configuration have no loops or redundant lines. Less sophisticated bridges leave this up to the system designer. With these bridges, you must set up your network so that there are no loops in it. More sophisticated bridges allow arbitrary topology, but disable links until no loops remain. This provides extra reliability. If a link fails, an alternative link will be turned on automatically. Bridges that work this way have protocol that allows them to detect when a unit must be disabled or reenabled, so that at any instant the set of active links forms a "spanning tree". If you require the extra reliability of redundant links, make sure that the bridges you use can disable and enable themselves in this way. There is currently no official standard for the protocol used among bridges, although there is a standard in the proposal stage. If you buy bridges from more than one vendor, make sure that their spanning-tree protocols will interoperate.

Gateways generally allow arbitrary network topologies, including loops and redundant links. Because gateways may have more than two interfaces, they must decide not only when to forward a packet, but where to send it next. They do this by maintaining a model of the entire network topology. Different routing techniques maintain models of greater or lesser complexity, and use the data with varying degrees of sophistication. Gateways that handle TCP/IP should generally support the two Internet standard routing protocols: RIP (Routing

Information Protocol) and EGP (External Gateway Protocol). EGP is a special-purpose protocol for use in networks where there is a backbone under a separate administration. It allows exchange of reachability information with the backbone in a controlled way. If you are a member of such a network, your gateway must support EGP. This is becoming common enough that it is probably a good idea to make sure that all gateways support EGP.

RIP is a protocol designed to handle routing within small to moderate size networks, where line speeds do not differ radically. Its primary limitations are:

- It cannot be used with networks where any path goes through more than 15 gateways. This range may be further reduced if you use an optional feature for giving a slow line a weight larger than one.

- It cannot share traffic between parallel lines (although some implementations allow this if the lines are between the same pair of gateways).

- It cannot adapt to changes in network load.

- It is not well suited to situations where there are alternative routes through lines of very different speeds.

- It may not be stable in networks where lines or gateways change a lot.

Some vendors supply proprietary modifications to RIP that improve its operation with EGP or increase the maximum path length beyond 15, but do not otherwise modify it very much. If you expect your network to involve gateways from more than one vendor, you should generally require that all of them support RIP, since this is the only routing protocol that is generally available. If you expect to use a more sophisticated protocol in addition, the gateways must have some ability to translate between their own protocol and RIP. However for very large or complex networks, there may be no choice but to use some other protocol throughout.

More sophisticated routing protocols are possible. The primary ones being considered today are cisco System's IGRP, and protocols based on the SPF (shortest-path first) algorithms. In general these protocols are designed for larger or more complex networks. They are in general stable under a wider variety of conditions, and they can handle arbitrary combinations of line type and speed. Some of them allow you to split traffic among parallel paths, to get better overall throughput. Some newer technologies may allow the network to adjust to take into account paths that are overloaded. However at the moment I do not know of any commercial gateway that does this. (There are very serious problems with maintaining stable routing when this is done.) There are enough variations among routing technology, and it is changing rapidly enough, that you should discuss your proposed network topology in detail with all of the vendors that you are considering. Make sure that their technology can handle your topology, and can

support any special requirements that you have for sharing traffic ong parallel lines, and for adjusting topology to take into account ailures. In the long run, we expect one or more of these newer routing protocols to attain the status of a standard, at least on a de facto basis. However at the moment, there is no generally implemented routing technology other than RIP.

One additional routing topic to consider is policy-based routing. In general routing protocols are designed to find the shortest or fastest possible path for every packet. In some cases, this is not desired. For reasons of security, cost accountability, etc., you may wish to limit certain paths to certain uses. Most gateways now have some ability to control the spread of routing information so as to give you some administrative control over the way routes are used. Different gateways vary in the degree of control that they support. Make sure that you discuss any requirements that you have for control with all prospective gateway vendors.

5.3.4 Network management

Network management covers a wide variety of topics. In general it includes gathering statistical data and status information about parts of your network, and taking action as necessary to deal with failures and other changes. There are several things that a switch can do to make this process easier. The most basic is that it should have a way gathering and reporting statistics. These should include various cts of packet counts, as well as counts of errors of various kinds. This data is likely to be most detailed in a gateway, since the gateway classifies packets using the protocols, and may even respond to certain types of packet itself. However bridges and even buffered repeaters can certainly have counts of packets forwarded, interface errors, etc. It should be possible to collect this data from a central monitoring point.

There is now an official Internet approach to network monitoring. The first stages use a related set of protocols, SGMP and SNMP. Both of these protocols are designed to allow you to collect information and to make changes in configuration parameters for gateways and other entities on your network. You can run the matching interface programs on any host in your network. SGMP is now available for several commercial gateways, as well as for Unix systems that are acting as gateways. There is a limited set of information which any SGMP implementation is required to supply, as well as a uniform mechanism for vendors to add information of their own. By late 1988, the second generation of this protocol, SNMP, should be in service. This is a slightly more sophisticated protocol. It has with it a more complete set of information that can be monitored, called the MIB (Management Information Base). Unlike the somewhat ad hoc collection of SGMP variables, the MIB is the result of numerous committee deliberations involving a number of vendors and users. Eventually it is expected that there will be a TCP/IP equivalent of CMIS, the ISO network monitoring service. However CMIS, and its protocols, CMIP, are not

yet official ISO standards, so they are still in the experimental stages.

In general terms all of these protocols accomplish the same thing: They allow you to collect criticial information in a uniform way from all vendors' equipment. You send commands as UDP datagrams from a network management program running on some host in your network. Generally the interaction is fairly simple, with a single pair of datagrams exchanged: a command and a response. At the moment security is fairly simple. It is possible to require what amounts to a password in the command. (In SGMP it is referred to as a "session name", rather than a password.) More elaborate, encryption-based security is being developed.

You will probably want to configure the network management tools at your disposal to do several different things. For short-term network monitoring, you will want to keep track of switches crashing or being taken down for maintenance, and of failure of communications lines and other hardware. It is possible to configurate SGMP and SNMP to issue "traps" (unsolited messages) to a specified host or list of hosts when some of these critical events occur (e.g. lines up and down). However it is unrealistic to expect a switch to notify you when it crashes. It is also possible for trap messages to be lost due to network failure or overload. Thus you should also poll your switches regularly to gather information. Various displays are available, including a map of your network where items change color as their status changes, and running "strip charts" that show packet rates and other items through selected switches. This software is still in its early stages, so you should expect to see a lot of change here. However at the very least you should expect to be notified in some way of failures. You may also want to be able to take actions to reconfigure the system in response to failures, although security issues make some mangers nervous about doing that through the existing management protocols.

The second type of monitoring you are likely to want to do is to collect information for use in periodic reports on network utilization and performance. For this, you need to sample each switch perodically, and retrieve numbers of interest. At Rutgers we sample hourly, and get the number of packets forwarded for IP and DECnet, a count of reloads, and various error counts. These are reported daily in some detail. Monthly summaries are produced giving traffic through each gateway, and a few key error rates chosen to indicate a gateway that is being overloaded (packets dropped in input and output).

It should be possible to use monitoring techniques of this kind with most types of switch. At the moment, simple repeaters do not report any statistics. Since they do not generally have processors in them, doing so would cause a major increase in their cost. However it should be possible to do network management for buffered repeaters, bridges, and gateways. Gateways are the most likely to contain sophisticated network management software. Most gateway vendors that handle TCP/IP are expected to implement the monitoring protocols described above. Many bridge vendors make some provisions for collecting performance data. Since bridges are not protocol-specific,

most of them do not have the software necessary to implement TCP/IP-based network management protocols. In some cases, monitoring can be done only by typing commands to a directly-attached console. (We have seen one case where it is necessary to take the bridge out of service to gather this data.) In other cases, it is possible to gather data via the network, but the monitoring protocol is ad hoc or even proprietary.

Except for very small networks, you should probably insist that all of the devices on your network collect statistics and provide some way of querying them remotely. In the long run, you can expect the most software to be available for standard protocols such as SGMP/SNMP and CMIS. However proprietary monitoring tools may be sufficient as long as they work with all of the equipment that you have.

5.3.5 A final evaluation

Here is a summary of the places where each kind of switch technology is normally used:

- Repeaters are normally confined to a single building. Since they provide no traffic isolation, you must make sure that the entire set of networks connected by repeaters can carry the traffic from all of the computers on it. Since they generally provide no network monitoring tools, you will not want to use repeaters for a link that is likely to fail.

- Bridges and gateways should be placed sufficiently frequently to break your network into pieces for which the traffic volume is manageable. You may want to place bridges or gateways in places where traffic would not require them for network monitoring reasons.

- Because bridges must pass broadcast packets, there is a limit to the size network you can construct using them. It is probably a good idea to limit the network connected by bridges to a hundred systems or so. This number can be increased somewhat for bridges with good facilities for filtering.

- Because certain kinds of network misbehavior will be passed, bridges should be used only among portions of the network where a single group is responsible for diagnosing problems. You have to be crazy to use a bridge between networks owned by different organizations. Portions of your network where experiments are being done in network technology should always be isolated from the rest of the network by gateways.

- For many applications it is more important to choose a product with the right combination of performance, network management tools, and other features than to make the decision between bridges and gateways.

@section(Configuring Gateways)

This section deals with configuration issues that are specific to gateways. Gateways than handle TCP/IP are themselves Internet hosts. Thus the discussions above on configuring addresses and routing information apply to gateways as well as to hosts. The exact way you configure a gateway will depend upon the vendor. In some cases, you edit files stored on a disk in the gateway itself. However for reliability reasons most gateways do not have disks of their own. For them, configuration information is stored in non-volatile memory or in configuration files that are uploaded from one or more hosts on the network.

At a minimum, configuration involves specifying the Internet address and address mask for each interface, and enabling an appropriate routing protocol. However generally a few other options are desirable. There are often parameters in addition to the Internet address that you should set for each interface.

One important parameter is the broadcast address. As explained above, older software may react badly when broadcasts are sent using the new standard broadcast address. For this reason, some vendors allow you to choose a broadcast address to be used on each interface. It should be set using your knowledge of what computers are on each of the networks. In general if the computers follow current standards, a broadcast address of 255.255.255.255 should be used. However older implementations may behave better with other addresses, particularly the address that uses zeros for the host number. (For the network 128.6 this would be 128.6.0.0. For compatibility with software that does not implement subnets, you would use 128.6.0.0 as the broadcast address even for a subnet such as 128.6.4.) You should watch your network with a network monitor and see the results of several different broadcast address choices. If you make a bad choice, every time the gateway sends a routing update broadcast, many machines on your network will respond with ARP's or ICMP errors. Note that when you change the broadcast address in the gateway, you may need to change it on the individual computers as well. Generally the idea is to change the address on the systems that you can configure to give behavior that is compatible with systems that you can't configure.

Other interface parameters may be necessary to deal with peculiarities of the network it is connected to. For example, many gateways test Ethernet interfaces to make sure that the cable is connected and the transceiver is working correctly. Some of these tests will not work properly with the older Ethernet version 1 transceivers. If you are using such a transceiver, you would have to disable this keepalive testing. Similarly, gateways connected by a serial line normally do regular testing to make sure that the line is still working. There can be situations where this needs to be disabled.

Often you will have to enable features of the software that you want to use. For example, it is often necessary to turn on the network management protocol explicitly, and to give it the name or address of a host that is running software to accept traps (error messages).

Most gateways have options that relate to security. At a minimum, this may include setting password for making changes remotely (and the "session name" for SGMP). If you need to control access to certain parts of your network, you will also need to define access control lists or whatever other mechanism your gateway uses.

Gateways that load configuration information over the network present special issues. When such a gateway boots, it sends broadcast requests of various kinds, attempting to find its Internet address and then to load configuration information. Thus it is necessary to make sure that there is some computer that is prepared to respond to these requests. In some cases, this is a dedicated micro running special software. In other cases, generic software is available that can run on a variety of machines. You should consult your vendor to make sure that this can be arranged. For reliability reasons, you should make sure that there is more than one host with the information and programs that your gateways need. In some cases you will have to maintain several different files. For example, the gateways used at Rutgers use a program called "bootp" to supply their Internet address, and they then load the code and configuration information using TFTP. This means that we have to maintain a file for bootp that contains Ethernet and Internet addresses for each gateway, and a set of files containing other configuration information for each gateway. If your network is large, it is worth taking some trouble to make sure that this information remains consistent. We keep master copies of all of the configuration information on a single computer, and distribute it to other systems when it changes, using the Unix utilities make and rdist. If your gateway has an option to store configuration formation in non-volatile memory, you will eliminate some of these gistical headaches. However this presents its own problems. The contents of non-volatile memory should be backed up in some central location. It will also be harder for network management personnel to review configuration information if it is distributed among the gateways.

Starting a gateway is particularly challenging if it loads configuration information from a distant portion of the network. Gateways that expect to take configuration information from the network generally issue broadcast requests on all of the networks to which they are connected. If there is a computer on one of those networks that is prepared to respond to the request, things are straightforward. However some gateways may be in remote locations where there are no nearby computer systems that can support the necessary protocols. In this case, it is necessary to arrange for the requests to be routed back to network where there are appropriate computers. This requires what is strictly speaking a violation of the basic design philosophy for gateways. Generally a gateway should not allow broadcasts from one network to pass through to an adjacent network. In order to allow a gateway to get information from a computer on a different network, at least one of the gateways in between will have to be configured to pass the particular class of broadcasts used to retrieve this information. If you have this sort of configuration, you should test the loading process regularly. It is not unusual to find that gateways do not come up after a power failure because someone changed the configuration of another gateway

and made it impossible to load some necessary information.

5.4 Configuring routing for gateways

The final topic to be considered is configuring routing. This is more complex for a gateway than for a normal host. Most Internet experts recommend that routing be left to the gateways. Thus hosts may simply have a default route that points to the nearest gateway. Of course the gateways themselves can't get by with this. They need to have complete routing tables.

In order to understand how to configure a gateway, we have to look in a bit more detail at how gateways communicate routes. When you first turn on a gateway, the only networks it knows about are the ones that are directly connected to it. (They are specified by the configuration information.) In order to find out how to get to more distant parts of the network, it engages in some sort of "routing protocol". A routing protocol is simply a protocol that allows each gateway to advertise which networks it can get to, and to spread that information from one gateway to the next. Eventually every gateway should know how to get to every network. There are different styles of routing protocol. In one common type, gateways talk only to nearby gateways. In another type, every gateway builds up a database describing every other gateway in the system. However all of the protocols have some way for each gateway in the system to find out how to get to every destination.

A metric is some number or set of numbers that can be used to compare routes. The routing table is constructed by gathering information from other gateways. If two other gateways claim to be able to get to the same destination, there must be some way of deciding which one to use. The metric is used to make that decision. Metrics all indicate in some general sense the "cost" of a route. This may be a cost in dollars of sending packets over that route, the delay in milliseconds, or some other measure. The simplest metric is just a count of the number of gateways along the path. This is referred to as a "hop count". Generally this metric information is set in the gateway configuration files, or is derived from information appearing there.

At a minimum, routing configuration is likely to consist of a command to enable the routing protocol that you want to use. Most vendors will have a prefered routing protocol. Unless you have some reason to choose another, you should use that. The normal reason for choosing another protocol is for compatibility with other kinds of gateway. For example, your network may be connected to a national backbone network that requires you to use EGP (exterior gateway protocol) to communicate routes with it. EGP is only appropriate for that specific case. You should not use EGP within your own network, but you may need to use it in addition to your regular routing protocol to communicate with a national network. If your own network has several different types of gateway, then you may need to pick a routing protocol that all of them support. At the moment, this is likely to

e RIP (Routing Information Protocol). Depending upon the complexity your network, you could use RIP throughout it, or use a more sophisticated protocol among the gateways that support it, and use RIP only at the boundary between gateways from different vendors.

Assuming that you have chosen a routing protocol and turned it on, there are some additional decisions that you may need to make. One of the more basic configuration options has to do with supplying metric information. As indicated above, metrics are numbers which are used to decide which route is the best. Unsophisticated routing protocols, e.g. RIP, normally just count hops. So a route that passes through 2 gateways would be considered better than one that passes through 3. Of course if the latter route used 1.5Mbps lines and the former 9600 bps lines, this would be the wrong decision. Thus most routing protocols allow you to set parameters to take this sort of thing into account. With RIP, you would arrange to treat the 9600 bps line as if it were several hops. You would increase the effective hop count until the better route was chosen. More sophisticated protocols may take the bit rate of the line into account automatically. However you should be on the lookout for configuration parameters that need to be set. Generally these parameters will be associated with the particular interface. For example, with RIP you would have to set a metric value for the interface connected to the 9600 bps line. With protocols that are based on bit rate, you might need to specify the speed of each line (if the gateway cannot figure it out automatically).

Most routing protocols are designed to let each gateway learn the topology of the entire network, and to choose the best possible route for each packet. In some cases you may not want to use the "best" route. You may want traffic to stay out of a certain portion of the network for security or cost reasons. One way to institute such controls is by specifying routing options. These options are likely to be different for different vendors. But the basic strategy is that if the rest of the network doesn't know about a route, it won't be used. So controls normally take the form of limiting the spread of information about routes whose use you want to control.

Note that there are ways for the user to override the routing decisions made by your gateways. If you really need to control access to a certain network, you will have to do two separate things: Use routing controls to make sure that the gateways use only the routes you want them to. But also use access control lists on the gateways that are adjacent to the sensitive networks. These two mechanisms act at different levels. The routing controls affect what happens to most packets: those where the user has not specified routing manually. Your routing mechanism must be set up to choose an acceptable route for them. The access control list provides an additional limitation which prevents users from supplying their own routing and bypassing your controls.

For reliability and security reasons, there may also be controls to allow you to list the gateways from which you will accept information. It may also be possible to rank gateways by priority. For example, you might decide to listen to routes from within your own organization

before routes from other organizations or other parts of the organization. This would have the effect of having traffic use internal routes in preference to external ones, even if the external ones appear to be better.

If you use several different routing protocols, you will probably have some decisions to make regarding how much information to pass among them. Since multiple routing protocols are often associated with multiple organizations, you must be sure to make these decisions in consultation with management of all of the relevant networks. Decisions that you make may have consequences for the other network which are not immediately obvious. You might think it would be best to configure the gateway so that everything it knows is passed on by all routing protocols. However here are some reasons why you may not want to do so:

- The metrics used by different routing protocols may not be comparable. If you are connected to two different external networks, you want to specify that one should always be used in preference to the other, or that the nearest one should be used, rather than attempting to compare metric information received from the two networks to see which has the better route.

- EGP is particularly sensitive, because the EGP protocol cannot handle loops. Thus there are strict rules governing what information may be communicated to a backbone that uses EGP. In situations where EGP is being used, management of the backbone network should help you configure your routing.

- If you have slow lines in your network (9600 bps or slower), you may prefer not to send a complete routing table throughout the network. If you are connected to an external network, you may prefer to treat it as a default route, rather than to inject all of its routing information into your routing protocol.

ERVICES	PORT #	ALIASIS	COMMENTS
echo	7/tcp		
echo	7/udp		
discard	9/tcp	sink null	
discard	9/udp	sink null	
systat	11/tcp	users	
daytime	13/tcp		
daytime	13/udp		
netstat	15/tcp		
qotd	17/tcp	quote	
chargen	19/tcp	ttytst source	
chargen	19/udp	ttytst source	
ftp	21/tcp		
telnet	23/tcp		
smtp	25/tcp	mail	
time	37/tcp	timserver	
time	37/udp	timserver	
rlp	39/udp	resource	# resource location
nameserver	42/tcp	name	# IEN 116
whois	43/tcp	nicname	
domain	53/tcp	nameserver	# name-domain server
domain	53/udp	nameserver	
mtp	57/tcp		# deprecated
tftp	69/udp		
rje	77/tcp	netrjs	
finger	79/tcp		
link	87/tcp	ttylink	
supdup	95/tcp		
hostnames	101/tcp	hostname	# usually from sri-nic
#csnet-cs	105/?		
pop	109/tcp	postoffice	
sunrpc	111/tcp		
sunrpc	111/udp		
auth	113/tcp	authentication	
sftp	115/tcp		
uucp-path	117/tcp		
nntp	119/tcp	readnews untp	# USENET News Transfer Protocol
snmp	161/udp		
#			
# APPLE Net Services			
#			
at-rtmp	201/udp		# udp rtmp
at-nbp	202/udp		# udp: nbp
at-echo	204/udp		# udp: echo
at-zis	206/udp		# udp: zip
at-zip	206/udp		
#			
#UNIX Specific services			

#			
exec	512/tcp		
biff	512/udp	comsat	
login	513/tcp		
who	513/udp	whod	
shell	514/tcp	cmd	# no passwords used
syslog	514/udp		
printer	515/tcp	spooler	# line printer spooler
talk	517/udp		
ntalk	518/udp		
efs	520/tcp		# for LucasFilm
route	520/udp	router routed	
timed	525/udp	timeserver	
tempo	526/tcp	newdate	
courier	530/tcp	rpc	
conference	531/tcp	chat	
netnews	532/tcp	readnews	
netwall	533/udp		#-for emergency broadcasts
uucp	540/tcp	uucpd	# uucp daemon
remotefs	556/tcp	rfs_server rfs	# Brunhoff remote filesystem
rmotd	569/tcp	uscbnews	
ingreslock	1524/tcp		

Appendix C

IP Addresses and Subnet Masks

ARPAnet was an early wide area network that connected host computers and terminal servers. The developers of ARPAnet set up procedures to allocate addresses to stations and to create voluntary standards for the network. As Local Area Networks proliferated, many hosts became gateways interconnecting local networks. A network protocol allowing these networks to work together was developed and called IP (Internet Protocol). Over time, other groups such as NASA and the Department of Defense created long-haul IP networks. IP enabled these groups to work together. The collection of these interoperating networks is the Internet. Informational Sciences Institute (ISI) does much of the research, standardization, and allocation work of the Internet. SRI International provides information services.

If you plan to connect your Series 4000s to the Internet, you must get a unique IP address for your local network before the local network can be connected to the Internet. Contact the following organization:

DDN Network Information Center
SRI International, Room EJ291
333 Ravenswood Avenue
Menlo Park, CA 94025
(800) 235-3155 or (415) 859-3695

Because an IP address is a 32-bit (4-octet) number, a large number of addresses are possible. IP addresses are commonly expressed in decimal dot notation as field1.field2.field3.field4, with each field being decimal numbers from 0 to 255 (e.g., 15.0.15.0, which in binary form equals 00001111 00000000 00001111 00000000).

Each bridge, router, and network station that uses IP for communication must have at least one unique IP address. The Series 4000 doesn't need an IP address to do bridge operations. However, the Series 4000's use of SNMP, which enables you to issue commands to configure or monitor a remote Series 4000, requires the Series 4000 to have an IP address because SNMP runs on top of IP. Each new Series 4000 has a preset IP address and subnet mask you can change to suit your network (described later). This preset IP address is the address of a software function (called the bridge port or B1 port) in the Series 4000.

When the Series 4000 is used as a router, it must be assigned at least one unique IP address for each network port (J1, J2, J3) to be used. Typically each (router) port has one IP address associated with it, although each port can have more than one.

For you to issue commands to a remote Series 4000, your Series 4000 must have a path to it, either directly or through intermediate routers or bridges. You must also know the IP address of any one of the Series 4000's ports to enable you to log into it. You can get any or all Series 4000 port IP addresses from the network administrator responsible for it, or by having someone at the remote Series 4000 issue a **display ip network table** command from its console terminal. Typically, the Ethernet port's IP address is the most likely to be readily known by the network administrator of the remote network.

When a Series 4000 receives a packet destined for an IP address, it must know whether it can give the packet directly to the destination, or whether it needs to give the packet to a router (or "gateway") that will forward the packet. The IP address contains enough information for a router to make this decision. If your Series 4000 sends data to a station on a remote network accessible through a router, you also need to obtain the IP address of that router. If the destination station is several routers away, you need only obtain the IP address of the nearest router (to your Series 4000) on the way there.

Class A, B, and C Networks

IP addresses are organized into Class A, B, and C networks. For a Class A network, the first (leftmost) field specifies the network number and Class. The remaining three fields specify the host (station) number and subnet address (if subnetworks are being used). The first field can be only from 1 to 126. 127 is reserved as a loopback address. This type of address scheme is good for a topology having a few large networks with very many stations.

For a Class B network, the first two (leftmost) fields specify the network number and class. The other two fields specify the host number and subnet address (if subnetworks are being used). The first field can be from 128 to 191. The second field can be

from 1 to 254. This type of address scheme is good for a topology having many networks and many stations.

For a Class C network, the first three fields specify the network number and class. The other field specifies the host number. The first field can be from 192 to 223. The second field can be from 0 to 255. The third field can be from 1 to 254. This scheme is good for a topology with many small networks having relatively few stations. The following table shows the ranges of the network portion of the addresses allowed for the three classes.

Class	Number
A	1 through 126
B	128.1 through 191.254
C	192.0.1 through 223.255.254

The binary value of the leftmost field defines the address format as follows:

Class A addresses have the leftmost bit equal to 0.

Class B addresses have the leftmost two bits equal to 1 0.

Class C addresses have the leftmost three bits equal to 1 1 0.

All Class A networks have already been allocated for the Internet, so you must choose between Class B and C when applying for an IP address. In the past, organizations needing many network addresses requested as many individual addresses as were needed (usually Class C), because much of the software available (notably 4.2 UNIX BSD) did not support subnetted addresses. Information on how to reach a remote network ("routing information") is stored in Internet gateways and routers. Some gateways and packet switches are limited to storing and exchanging routing information for about 300 networks, so Internet management suggests that each organization uses no more than two network numbers on the Internet. If your organization expects to be constrained by this, you can use subnetting.

Subnetting

Subnetting allows you to use one address on the Internet and a subset of addresses within your organization. To do this, you must define a mask at each of your organization's stations and routers that distinguishes between the network and host portions of the address. For example, if your organization needs two networks internally and has the roughly 64K addresses beginning 128.64.x.x (a Class B address) allocated to it, you could allocate 128.64.1.x to one part of the organization and 128.64.2.x to another. ("x" stands for any number from 1 to 254. The numbers 0 and 255 are reserved for network addresses and broadcast addresses.) Stations on the Internet could reach stations on your organization by way of the network portion of the address 128.64. The Internet treats your organization's two blocks of addresses 128.64.1.x and 128.64.2.x as one network. (The rightmost two octets of a Class B address represent a host address. The Internet therefore ignores these octets when passing packets to the destination network.) In the Series 4000, you can define subnet masks for Internet addresses by commands from a console terminal. For other systems (e.g., UNIX), subnet masks are defined in various ways.

Subnet routing requires your organization's systems to interpret IP addresses slightly differently. Instead of only two fields (network and host) in the IP address, your systems must interpret the address to have three fields: network, subnetwork (also called a subnet), and host. Your system must interpret some of the bits in the host field as the subnetwork part of the IP address. Each system must use a subnet mask to do this.

A subnet mask is a 32-bit number that corresponds bit-for-bit with the 32-bit IP address. For each bit set to 1 in the subnet mask, the corresponding bit position in the IP address is interpreted as part of the network and subnetwork address. If the subnet mask bit position is part of the host field and is set to 1, the corresponding bit in the IP address is interpreted as part of the subnetwork address. If the subnet mask bit position is part of the host field and is set to 0, the corresponding bit in the IP address is interpreted as part of the host address.

For any IP address you have been assigned, each bit in the first (left-most) field of the subnet mask must be set to one (decimal 255, binary 11111111), because the first field is always a network address field, regardless of whether you are using subnetworks. The subnet mask 255.255.255.255 is not used because all bits of

the corresponding IP address would be interpreted as the network address, leaving no bits for the host address. The subnet mask 0.0.0.0 is not used because the entire IP address would be interpreted as the host address, leaving no bits for the network address.

Typically, entire 8-bit fields are set to all ones or all zeroes. This makes it easier for system managers to distinguish among the network, subnet, and host fields. Using the values 255 and 0 allows the fields to be divided evenly between the subnet and host parts of the IP address. Values other than 255 and 0 do not divide evenly, making the host and subnet parts of the address harder to understand.

The first field of the subnet mask is always 255, so the system can interpret the network number. The fourth field is usually 0, so the system can interpret the host address. The second and third fields are usually 255 or 0, depending on how the IP address is to be interpreted. So the subnet masks most often used are as follows:

255.255.255.0
255.255.0.0
255.0.0.0

Example of Class B Subnet Mask

Suppose the Network Information Center allocates your organization the IP addresses 128.64.x.x. If your organization has two networks for which you want to segregate data traffic, you could assign one network the IP addresses 128.64.1.x and the other network the addresses 128.64.2.x. Applying a subnet mask of 255.255.255.0 to the IP addresses at all stations enables the stations within the organization to treat the two address groups as separate entities. Suppose you have a station with IP address 128.64.1.4 on one network, and a station with IP address 128.64.2.7 on another network. As shown in the following figure, when the mask 255.255.255.0 is applied (ANDed) to the address 128.64.1.4, the result is 128.64.1.0. Similarly, when the mask 255.255.255.0 is applied to the address 128.64.2.7, the result is 128.64.2.0. The two addresses are viewed as residing on distinct address groups within the organization.

```
                          |   Network Portion     |    Host Portion
                          |                       |
Address (128.64.1.4):     | 10000000  01000000    | 00000001  00000100
Mask (255.255.255.0):     | 11111111  11111111    | 11111111  00000000
--------------------      |-----------------------|--------------------
Result  (128.64.1.0):     | 10000000  01000000    | 00000001  00000000

                          |   Network Portion     |    Host Portion
                          |                       |
Address (128.64.2.7):     | 10000000  01000000    | 00000010  00000111
Mask (255.255.255.0):     | 11111111  11111111    | 11111111  00000000
--------------------      |-----------------------|--------------------
Result  (128.64.2.0):     | 10000000  01000000    | 00000010  00000000
```

The first two digits of 128 in binary notation are 10 (128 in binary is 10000000). So the IP addresses are on a Class B network. The first two fields of the IP addresses (128.64) are the network part. The third field is interpreted as the subnet part, and the fourth field is interpreted as the host part. So the subnet mask and IP addresses used in this example allow 16 bits to identify the network, 8 bits to identify the subnet, and 8 bits to identify the host. This means you can have 256 subnets and 254 hosts. (Two numbers in the host portion are reserved for special usage: The host number 00000000 is used to indicate the network address, and the host number 11111111 is reserved for use as a broadcast address.)

Examples of Commonly Used Subnet Masks

Table 1 shows some commonly used examples of Class B subnet masks. Table 2 shows the only subnet masks allowed on a Class C network. In the subnet masks of the following tables, bits equal to 1 make up the network part, and bits equal to 0 make up the the host part. Not all host numbers are allowed in the IP addresses used with the following subnet masks. Two numbers in the host portion are reserved for special usage: The host number 00000000 is reserved to indicate the network address, and the number 11111111 is reserved for use as a broadcast address. The tables that follow show the maximum number of subnets and hosts allowed for each type of mask.

Table 1.
Examples of Class B - Subnet Masks

Subnet Mask	Subnets Allowed	Hosts Allowed
255.255.192.0 11111111 11111111 11000000 00000000	2	16,382
255.255.224.0 11111111 11111111 11100000 00000000	6	8190
255.255.240.0 11111111 11111111 11110000 00000000	14	4094
255.255.248.0 11111111 11111111 11111000 00000000	30	2046
255.255.252.0 11111111 11111111 11111100 00000000	62	1022
255.255.254.0 11111111 11111111 11111110 00000000	126	510
255.255.255.0 11111111 11111111 11111111 00000000	254	254

Table 2.
Class C Subnet Masks

Subnet Mask	Subnets Allowed	Hosts Allowed
255.255.255.192 11111111 11111111 11111111 11000000	2	62
255.255.255.224 11111111 11111111 11111111 11100000	6	30
255.255.255.240 11111111 11111111 11111111 11110000	14	14
255.255.255.248 11111111 11111111 11111111 11111000	30	6
255.255.255.252 11111111 11111111 11111111 11111100	62	2

Series 4000 Preset IP Address and Subnet Mask

Each new Series 4000 has a preset IP address and subnet mask you can change to suit your network. This preset IP address is the address of a software function (called the bridge port or B1 port) in the Series 4000. The preset address is 126.x.y.z where x, y, and z are the decimal equivalent of each byte of the rightmost three bytes of the MAC address. For example, a Series 4000 with MAC address 08:00:03:41:03:F9 has the preset IP address 126.65.3.249. (Hexadecimal 41, 03, and F9 equal decimal 65, 3, and 249 respectively.) The preset subnet mask of each Series 4000 is 255.0.0.0.

To remotely set up a Series 4000, you need to know one of its IP addresses to be able to log into it. You can figure out the bridge port IP address of a Series 4000 if you know its serial number, because you can deduce the IP address from its MAC address, and you can deduce the MAC address from its serial number. (Refer to the section "MAC Addresses" earlier in this chapter.) An easier way to find a usable IP address is to have someone at that site issue a **display ip network table** command, then tell you one of the addresses.

Appendix D

ACS XNS Software
Technical Overview
For routing over the StreamLine 4100 & StreamLine 4200 Platforms

Protocol Overview
XNS (Xerox Network Systems) is a family of protocols developed at Xerox PARC which provides the foundation of Xerox distributed systems. These protocols were designed to operate across a variety of communications media, including Ethernet, X.25 and point-to-point lines (Ethernet is the primary media). Various vendors of networking products (e.g., Novell, 3Com, Ungermann Bass, Banyan, Apollo) have adapted XNS for their use by incorporating proprietary extensions to the basic protocols and adding new protocols of their own.

It is gandalf's intention that the StreamLine Series 4000 will support all XNS derivatives which have a sufficiently large market. Since all of the vendors mentioned above have extended the basic protocol to accommodate their needs, support for these protocols will be incorporated as technical specifications can be acquired. The protocols which will be initially supported are **IDP (Xerox)** and **IPX (Novell)**. (For more information on Novell's IPX protocol refer to the August 15th issue of the Product Marketing Manager's Technical memos.)

Internet Datagram Protocol (IDP)
Internet Datagram Protocol (IDP) provides an unreliable datagram service to upper-level XNS protocols. IDP is IP-like (as in TCP/IP), but without options or fragmentation. XNS generally uses Ethernet with an Ethernet type of 0x0600 for transporting IDP datagrams.

Addressing
The 32-bit network number identifies a network within the internet, the 48-bit host number identifies a host within the internet (and not the subnetwork), and the 16-bit socket number identifies the source or destination of packets within a host. The host number is globally unique, and by convention it is the physical (MAC-level) address of the device which connects the host to the communications media. When a host is connected to more than one network, the physical address on each network must be set to the 48-bit host address. Therefore, the mapping between XNS network addresses and MAC-level addresses is trivial. In particular, ARP or an ARP-like protocol for address translation is not required.

The host number space is flat, as is the network number space. Since the host number uniquely identifies a particular host, the network number is redundant;

its only use is to assist in internetwork routing. Generally, network numbers are assigned serially, but may be any arbitrary 32-bit value. All routing decisions are based on the network number.

Packet Length

The length field is the length of the packet in bytes, including the IDP header and length field itself. The nominal packet length is 576 bytes. Processes are allowed to negotiate a larger packet size, but there are no built-in facilities or options to do this. Internetwork fragmentation and reassembly must be performed by the lower-layer protocols (e.g., X.25). Routers are free to discard packets which are too large to handle, and there is no reliable mechanism to discover the MTU supported along a given route (except by noticing error messages returned by the error protocol which give the maximum MTU for the node which generated the error).

Hop Count

The transport control byte contains a 4-bit hop count used to limit the lifetime of a packet. This field is initialized to zero and incremented by each router which forwards the packet. When hop count reaches a value of 16, packets are discarded. Maximum packet lifetime is estimated to be 60 seconds (4 seconds per hop). Maximum packet lifetime is bounded by the hop count and the requirement that the network device drives discard packets which have been on the transmit queue longer than 6 seconds.

Routing Protocols

The packet type field is used to identify the format of the data field of the internet packet. IDP does not interpret this field; it merely delivers an arriving packet to the appropriate socket, the client of which interprets the packet type. The following packet types are standard:

- **Routing Information Protocol (RIP)**
- **Echo Protocol**
- **Error Protocol**
- **Packet Exchange Protocol (PEP)**
- **Sequenced Packet Protocol (SPP)**

The echo and error protocols provide services similar to ICMP. The error protocol can report unreachable destinations, but has no redirect to alter routes. PEP is similar in capability to UDP with retransmission ("at least once" delivery), and SPP is similar to TCP ("always once" delivery).

XNS routing information protocol (RIP) formed the basis of the IP-based routing information protocol implemented in BSD Unix. Therefore, the two protocols are very similar.

- all routes are based on network number (i.e., no host or subnet routes)
- split horizon is not used
- no random skewing of gratuitous updates or random delay of triggered updates
- not limited to 512-byte maximum packet size.

Conclusion

XNS and IPX are employed on over 50% of all the PC LAN's installed world wide. The addition of these protocols into the StreamLine 4000 Series family of router products allows the customer to route IPX and XNS throughout the network. The StreamLine 4000 Series products will look to the network like an analogous XNS or IPX router.

Appendix E

ACS IPX Routing Software
Technical Overview
For routing over the StreamLine 4000 Series Platforms

Protocol Overview
XNS (Xerox Network Systems) is a family of protocols developed at Xerox PARC which provides the foundation of Xerox distributed systems. These protocols were designed to operate across a variety of communications media, including Ethernet, X.25 and point-to-point lines (Ethernet is the primary media). Various vendors of networking products (e.g., Novell, 3Com, Ungermann Bass, Banyan, Apollo) have adapted XNS for their use by incorporating proprietary extensions to the basic protocols and adding new protocols of their own. The purpose of this document is to describe **Novell's IPX** Routing Protocol.

Novell's Internet Packet Exchange (IPX)

Background
IPX is Novell's proprietary adaptation of IDP used in their NetWare LAN products (ELS NetWare, Advanced NetWare, SFT NetWare, etc.). Similarly, SPX is the Novell adaptation of Xerox SPP. IPX dominates the PC LAN market place; over 60% of the installed PC networks use NetWare as the LAN operating system (*PC Week, Business Section August 27, 1990*).

IPX is a Novell communication protocol which create, maintains, and ends connections between network devices (workstations, file servers, bridges, etc.). IPX addresses and routes outgoing data packets across a network. For returning data, IPX reads the assigned addresses and directs the data to the proper area within a workstation's or file server's operating system.

IPX is closely linked with a number of other programs and routines which help in the network data-transmission process. NetWare prepares data packets in a form understandable to the intended destination before handing them to IPX. IPX uses the services of the SPX protocol which monitors transmission to assure correct delivery.

SPX is a Novell communication protocol which monitors network transmissions to ensure successful delivery. SPX is derived from IPX using the Xerox Sequenced Packet Protocol. SPX enhances the IPX protocol by supervising data sent across the network. SPX can track data transmissions which consist of a series of separate packets. SPX also requests, and returns, acknowledgements from a communications partner ensuring successful data

delivery. If an acknowledgement request brings no response within a specified time, SPX will retransmit it. After a reasonable number of retransmissions fail to return a positive acknowledgement, SPX assumes the connection has failed and will warn the operator of the failure.

Terminology

Novell uses different terminology to define network components. For example, routers are referred to as ***bridges*** (they do not describe, and apparently do not use, MAC-level bridges), and bridges may be ***internal*** or ***external*** depending on whether they are implemented within the file server or work station. The StreamLine Series 4000 would be a ***dedicated external bridge*** using Novell terminology. Nevertheless, the proprietary NetWare protocols are reasonably close in function and format to their XNS counterparts. Therefore, only the differences are highlighted below:

The default encapsulation for IPX is 802.3 **without** 802.2. NetWare can be configured to run with Ethernet 2.0 encapsulation using an Ethernet type of 0x8137.

NetWare uses non-standard packet types. For example, RIP packets are transmitted with a packet type of 0 (XNS "unknown" packet type). Other undocumented packet types are used by upper-layer protocols (see paragraph below on internetwork broadcast). In particular, a packet type of 0x11 has been observed, which is referred to in one NetWare user's manual as "NetWare Control Protocol" and in another as "NetWare Core Protocol".

Hop Count & Timers

NetWare routing code implements the simple form of split horizon (i.e., without poisoned return) when generating gratuitous and triggered updates. Also, split horizon is employed when generating responses to specific requests (IP RIP returns all known routes assuming that a specific request may be a diagnostic query).

Route table entries with a hop count of infinity are never returned in a response packet.

The source network number may be zero, implying the local network number from which a RIP packet was received. NetWare work stations use a network number of zero until the correct value is learned from a routing update received from a router, therefore avoiding the need to configure this information.

NetWare limits the number of entries in a routing information update to 50 (there is room in a 576-byte packet for 68).

In addition to a hop-count metric, NetWare also uses a transit-time metric. The metric is defined as the transit time of a 576-byte packet in IBM PC timer units (18.21 timer units per second).

Multiple routes to the same destination network are maintained in the routing table and ranked according to minimum transit time first, minimum hop count second.

If a neighboring router reports a destination no longer reachable and an alternate route exists, the router knowing of an alternate route will advertise it if the route still exists 5 seconds later.

Updates are broadcast at 60-second intervals. The reclaim interval is the same as XNS (240 seconds).

Only triggered updates are sent on serial links. Such links are assumed to be reliable (i.e., update packets will not be dropped) and, therefore, updates only need to be sent when a change occurs. Route table entries received on a serial link are not aged.

Other Protocols

Some additional proprietary protocols have been defined. For example, a network management protocol (Diagnostic Services) has been implemented above SPX. Also, a resource location protocol **(Service Advertising Protocol (SAP))** has been implemented above IPX.

SAP is used by servers to advertise their existence, and by work stations to locate the servers. Routers participate by maintaining a dynamic cache of all known (i.e., advertised) servers, by distributing this information to other routers, and answering specific queries. he router must act as a surrogate for non-dedicated servers which operate as terminate-and-stay-resident programs. SAP is similar to RIP; instead of maintaining routes to destination networks, SAP maintains the address of servers which are referenced by name and/or type. As in IP RIP, a hop count is used as the metric in determining preference (no delay metric).

The IPX interlan broadcast takes the place of the directed broadcast (or multicast) supported by standard XNS. XNS explicitly prohibits a global broadcast.

Conclusion

XNS and IPX are employed on over 60% of all the PC LAN's installed world wide. The addition of these protocols into the StreamLine 4000 Series family of router products allows the customer to transparently route IPX throughout the Novell network. The StreamLine 4000 Series products will see the network as an analogous XNS or IPX router.

Appendix F

DECnet Routing Software
Technical Overview
For the StreamLine 4000 Series Platforms

Protocol Overview

What is DECnet?

Digital Equipment Corporation (DEC) has specified a model for connecting together the computers they manufacture that is known as the DIGITAL Network Architecture (DNA). DECnet is a family of software and hardware products that together provide an implementation of DNA. DECnet is by far the most popular protocol in use on Ethernets today, mainly due to DEC's use of the Ethernet as the preferred way to "front-end" their CPUs. Sites with DEC hardware account for the largest part of the installed base of Ethernets.

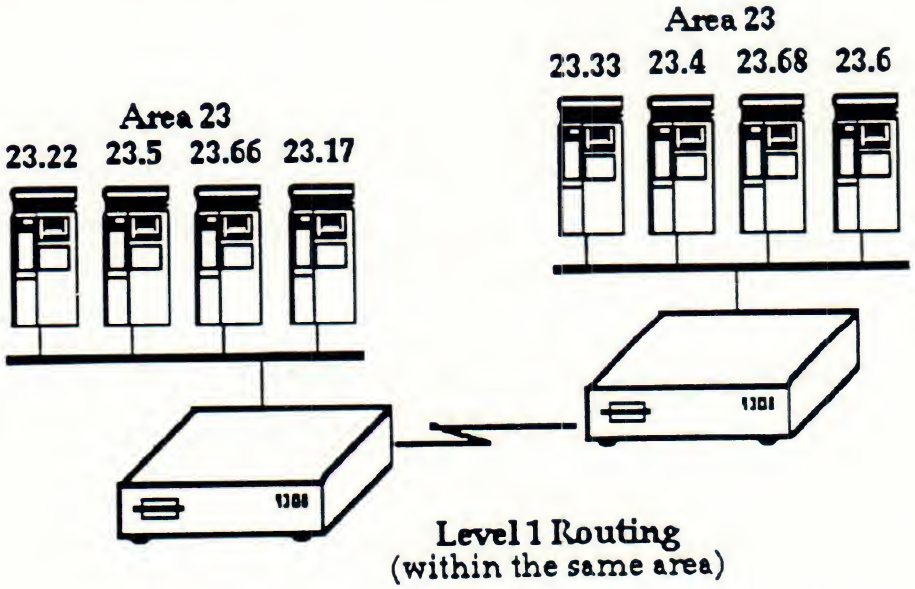
Terminology

DEC uses its own terminology to define network components. The following is a partial list of DEC defined hardware, protocols and values used by DEC in conjunction with DECnet Routing.

Adjacent-	A term used to describe End Nodes in the same Area and the interconnected routers outside of an Area.
Area-	A logical DECnet community made up of interconnected hosts all sharing the same area number.
Circuit-	A physical link between two Nodes.
Circuit Cost-	An administrative cost assigned to a circuit for route computation.
End Node-	A Device on a DECnet, such as a host, printer, terminal server or any other addressable device.
Router Node-	A DECnet router. The Router node may be a stand alone system, as is ACCs DECnet router, or it may be a host with 2 interface cards and the proper software.
Level 1 Router-	A Router Node that routes within a single area.
Level 2 Router-	A Router Node that routes within an Area and between Areas.

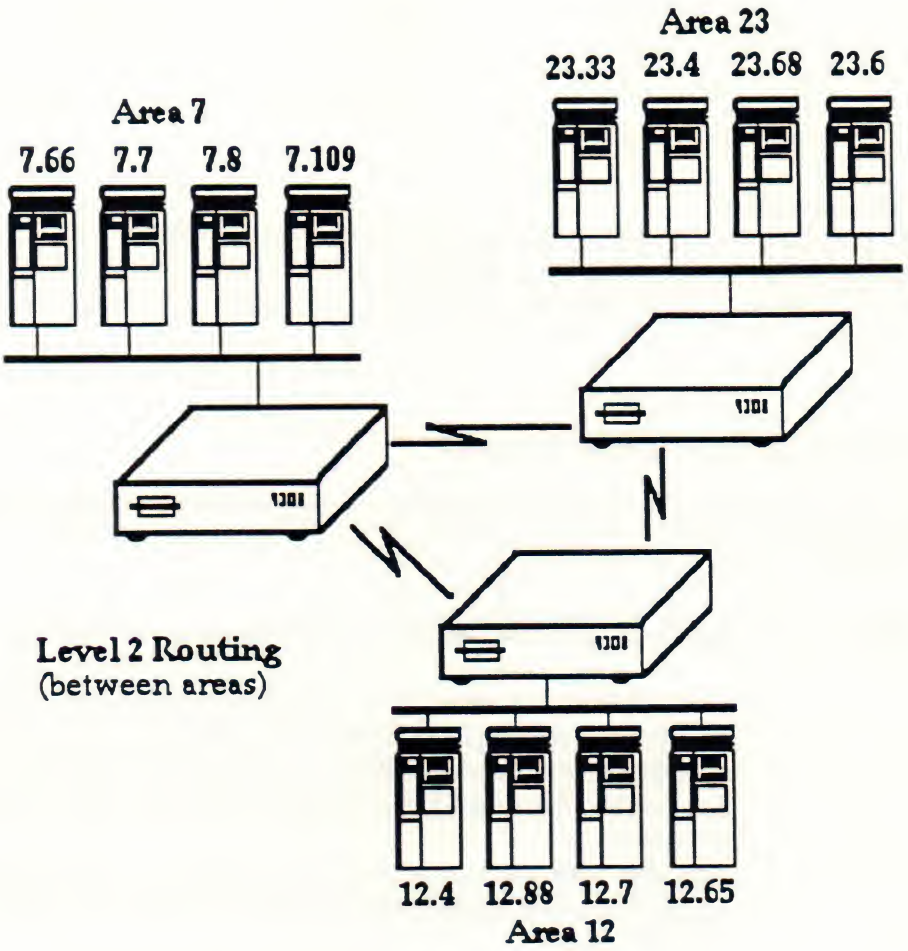
What is DECnet Level 1 Routing and DECnet Level 2 Routing?
In order to support large networks, DECnet employs a hierarchical routing scheme. In DECnet terminology, each computer system is called an "end-node". DECnet routers are also a type of "node", and each node belongs to an "area". A large network may consist of several areas. Routing within a single area is known as "level 1 routing", while routing between multiple, or more than one area is known as "level 2 routing". A level 1 router keeps track of routing within its own area, and also knows about the nearest level 2 router within its area. A level 2 router does everything a level 1 router does, and in addition, knows routes to other areas. Thus, a level 2 router serves as a gateway between areas. A maximum of 63 areas and 1023 nodes per area are supported by DECnet.

Level 1 Routing



In the above example; a single area (Area 23) is divided into two geographically separate Ethernets. The routers maintain tables which indicate nodes accessible through either router.

Level 2 Routing



In the above example; There are 3 geographically separate areas (Area 23, 7 & 12). The routers maintain tables indicating which nodes are accessed through a given router and which routes a packet must take to arrive at a given area.

Addressing
DEC uses a 16 bit field to identify a Node Address. The first 6 bits designate the Area number (from 1-63 areas are supported). The next 10 bits identify the particular Node within a given Area (from 1-1023 nodes are supported).

The MAC layer address is derived from the Node Address. DEC uses the AA:00:04 to identify the packet as DECnet Phase IV which invokes an algorithm to compute the Node Address in HEX for the remaining portion of the MAC layer address. The structure of the MAC layer address is as follows:

AA:00:04:	00:	xx:yy
DECnet Phase IV	*Reserved*	*Node Address*

Packet Length
The length field is the length of the packet in bytes, including header and length field itself. The minimum packet length is 64 bytes. Internetwork fragmentation and reassembly must be performed by the lower-layer protocols (e.g., X.25).

DECnet Ethernet Frame Format

Destination Address	Source Address	Type Field	Count	Data	Pad	FCS
48 bits	48 bits	16 bits	16 bits	8n bits	8m bits	32 bits

- Destination Address is the data link address of the receiving station. It can be of 3 types; Physical Address (unique to each station), Multicast Address (a multi-destination address), or Broadcast Address (defined as being sent to all stations on an Ethernet).
- Source Address is the data link address of the sending station.
- Type Field is denotes what kind of packet it is (such as DECnet, TCP/IP, etc.).
- Count is a 16 bit count of DATA bytes to follow.
- Data is the actual data of the packet.
- Pad is the amount of bytes of padding inserted to reach the minimum Ethernet packet length.

One of the most common DECnet Protocols is LAT (Local Area Transport) Server protocol. LAT has a protocol type of 6004. LAT protocol can not be routed. It can only be bridged. LAT packets are padded out to meet the minimum length requirement. The padding can be truncated and reinserted as needed to allow for compression across serial bridged links.

Hop Count
The Logical distance from one node to another node in a network is a "hop". The complete distance that a packet travels from source to destination is the "Path Length". Path Length is measures in hops. The maximum number of hops

the routing algorithm will forward is called "Maximum visits". The number is limited to 63 by the routing architecture.

Ethernet Router Hello Message

The Ethernet Router Hello message is used for initializing and monitoring of routers on an Ethernet. Each Ethernet router periodically broadcasts an Ethernet Router Hello message to all other nodes (both Routers and End Nodes on the Ethernet) using a multicast address. The messages contain a list of all routers on the Ethernet who have acknowledged the Hello messages.

Ethernet End Node Hello Message

Ethernet End Nodes periodically broadcast Ethernet End Node Hello messages to inform the routers of their status, whether up or down.

Conclusion

DECnet is employed on over 50% of all the System LAN's installed world wide. The addition of DECnet Routing into the StreamLine 4000 Series family of router and bridge products gives the customer the advantage of DECnet routing and MAC layer bridging (with LAT compression). The StreamLine 4000 Series products will look to the network like an analogous DECnet router and/or MAC layer bridge. The distinct advantage is the fact that the StreamLine 4000 Series products can do both functions concurrently.

Appendix G

AppleTalk Routing Software
Technical Overview
For the StreamLine 4000 Series Platforms

What is AppleTalk?
The AppleTalk Personal Network was designed by Apple Computers as a simple, inexpensive, and flexible way to interconnect computers, peripheral devices and servers. Simply put, it employes the same philosophy behind the development of the Apple Macintosh. Apple estimates that there are 2 million nodes installed on more than 250,000 networks. (NOTE: Apple's minimal "network" definition consists of a single MAC with an attached printer.)

The Series 4000 AppleTalk software is modeled after the Apple Internet Router. Each platform running the Series 4000 AppleTalk software is a full router. All StreamLine 4000 Series platforms running the Series 4000 AppleTalk software will interoperate with Apple Internet Routers and other AppleTalk-compliant products. The Series 4000 AppleTalk software is modeled as a *half router.*

The StreamLine 4000 AppleTalk supports both AppleTalk Phase 1 and AppleTalk Phase 2.

The StreamLine 4000 AppleTalk supports both EtherTalk and TokenTalk.

The following StreamLine 4000 link-level protocols are supported:

LAN	**WAN**
Ethernet	LAPB
Token Ring	Multilink LAPB
	X.25

The following AppleTalk protocols are be supported and are implemented for operation as an internet router:

ELAP	EtherTalk Link Access Protocol
AARP	AppleTalk Address Resolution Protocol
TLAP	TokenTalk Link Access Protocol
DDP	Datagram Delivery Protocol
RTMP	Routing Table Maintenance Protocol
AEP	AppleTalk Echo Protocol
ATP	AppleTalk Transaction Protocol
NBP	Name Binding Protocol
ZIP	Zone Information Protocol

Terminology

Apple uses its own terminology to define network components. The following is a partial list of Apple defined hardware, protocols and values used by Apple in conjunction with AppleTalk Routing.

Bridge- A router between two local AppleTalk network. (also referred to as a **Bridge-Node**)

EtherTalk- AppleTalk datagrams encapsulated within Ethernet frames.

Gateway- A device that connects a LocalTalk network to other networks such as EtherTalk, TokenTalk, Novell, etc.

Internetting- The connecting of several AppleTalk networks via routers. (The term **internet** refers to an AppleTalk network consisting of several smaller LocalTalk, EtherTalk or TokenTalk networks which are interconnected.)

LocalTalk- Apple Computer's proprietary physical cabling scheme. It is basically a twisted pair Ethernet using CSMA/CD. It's maximum throughput is 230.5 kilobits per second over a maximum of 300 meters.

Node- A device on an AppleTalk network, such as MACs, printers and servers.

Router- A device which routes datagrams between two remote AppleTalk networks.

TokenTalk- AppleTalk datagrams encapsulated within Token Ring frames.

Zone- An arbitrary subset of all AppleTalk networks in an internet. A particular network can only belong to one zone. (Zones are identified by a string which can not exceed 32 characters.)

Overview

AppleTalk is structured around the 7 layer ISO model. The Network, Datalink and Physical Layers are very similar in structure to the TCP/IP or XNS model. In fact they use many of the same terms and components as these common protocol stacks. For instance, the smallest addressable AppleTalk data unit is called a datagram just as it is in IP and XNS.

AppleTalk allows resources such as file servers, modems, printers and other peripheral devices, to be shared between nodes within a workgroup. Each

LocalTalk network can have up to 32 nodes or devices interconnected within that network.

Large numbers of AppleTalk networks can be connected together to form a complex internet.

Apple has defined the terms "Bridge" and "Internet Router" a little differently then the TCP/IP community. For AppleTalk networks the term "Bridge" refers to a local router. It is a device connected to two local AppleTalk networks and it routes between them. The term "Internet Router" refers to a remote router. It is a device connected to two remote AppleTalk networks, these networks have a WAN in between them.

AppleTalk provides a mechanism to dynamically assign an address when a node is initialized, so nodes do not need to be manually configured each time and thus keeping with Apple's plug-and-play philosophy. This is handled by AppleTalk's Address Resolution Protocol (AARP). AARP also provides the same functions as IP ARP to resolve hardware and protocol addresses mapping.

The Datagram Delivery Protocol (DDP) handles forwarding of AppleTalk packets based on lookups in a route table. It also recognizes packets which are addressed to its local ports and socket numbers. This is a best effort service and packets may be lost. It is very similar to Internet Protocol as used in TCP/IP.

AppleTalk's Routing Table Maintenance Protocol (RTMP) accumulates and broadcasts routes on a periodic basis. This eventually results in the routers obtaining a route to each network in the connected internet including routes over remote links.

The Name Binding Protocol (NBP) is needed since node addresses are dynamic and users require some means to identify nodes in order to communicate with them. NBP allows a node to register a name which users associate with the particular node. This functions allows a node on a AppleTalk network to find a printer or server within its network or within the internet.

The Zone Information Protocol (ZIP) provides network to zonename mapping in conjunction with NBP name registration and lookups.

AppleTalk Echo Protocol (AEP) provides an echo function to verify that a particular node is up and can be used to determine a roundtrip time for packet delivery.

The AppleTalk Transaction Protocol (ATP) is used by ZIP for delivery of specific packet types. This is a reliable protocol which guarantees delivery of datagrams.

Phase 1 AppleTalk -

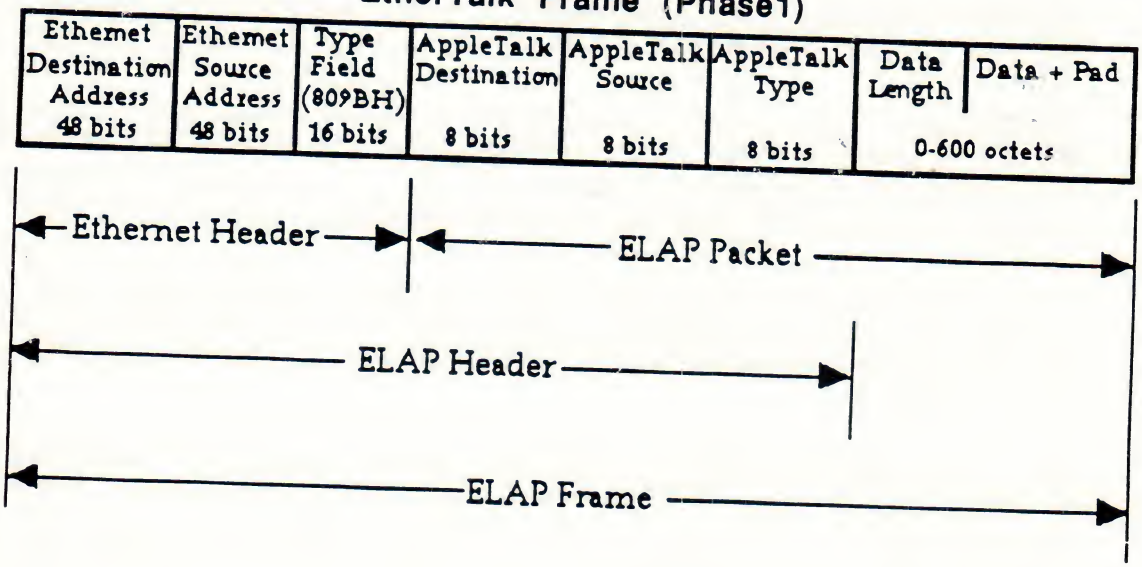
Phase 1 AppleTalk has a major limitation, the addressing scheme supports no more than 254 nodes per network (LocalTalk drops this further to a maximum of 32 devices because of physical cable limitations).

AppleTalk addresses consist of the following:

- a network number (2 bytes)
- a node id number (1 byte)
- a socket number (1 byte)

The network number identifies an individual cable. The node id uniquely identifies a node on the given network. Thus the length of the node id creates the 254 limit (0 and 255 are reserved). Each entity on the node is uniquely identified by a socket number.

EtherTalk Frame (Phase1)



Phase 2 AppleTalk -

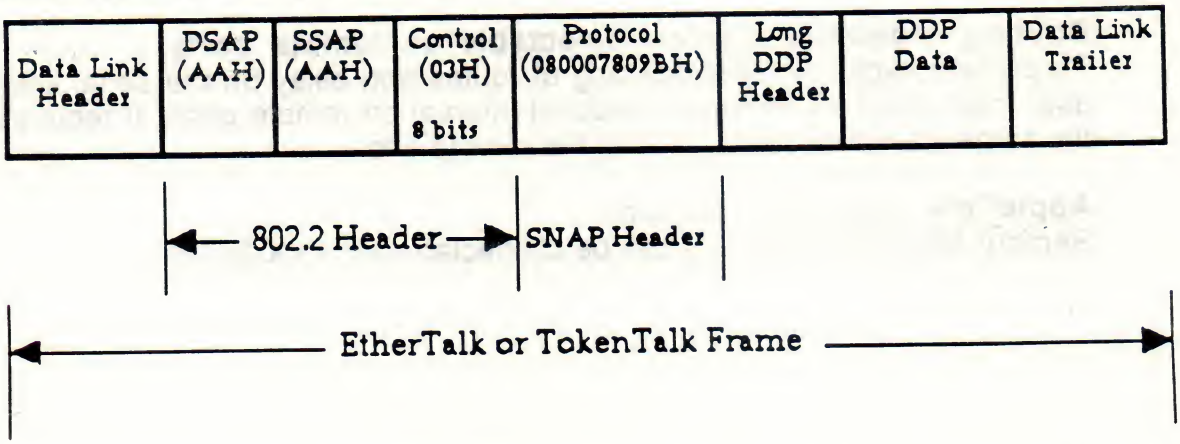
Phase 2 relieves the addressing limitation by allowing multiple network numbers on a single cable, thus the limitation is expanded to 254 times the number of networks assigned to the cable.

A new Ethernet broadcast address was defined to relieve non-AppleTalk nodes from receiving and filtering out AppleTalk broadcasts (IEEE Protocol ID for AppleTalk over Ethernet, EtherTalk is **809B**, AARP multicast protocol type is **80F3**.). A multi-cast address based on zonenames has also been incorporated to reduce traffic to zone-wide rather than cable-wide broadcasts.

Phase 2 allows for easy encapsulation of datagrams into either Ethernet frames or Token Ring frames by using 802.3 or 802.5 with SNAP headers.

Phase 2 has additional changes which affect existing Phase 1 packet formats and define additional packet types.

EtherTalk Frame (Phase 2)



Unfortunately, these changes make Phase 1 and Phase 2 incompatible. This is why Apple offers a special router upgrade kit to allow phase 1 and phase 2 nodes on the same network. This upgrade kit requires all routers to be upgraded and was intended as strictly temporary until all nodes can be upgraded. It also imposes the 254 node limit to make the two compatible.

ACC Implementation Features

The Series 4000 routing software will connect remote Ethernets via the remote links. It does not support direct connection of LocalTalk cables. LocalTalk requires attachment to Ethernet by either a Shiva FastPath or an Apple Internet router (Apple software on a dedicated Mac).

Multi Protocol Routing.
Simultaneous routing of IP, DECnet, Novell's IPX, XNS and AppleTalk is supported as well as concurrent bridging.

SNMP Network management.
Management of all ACC products are via SNMP. The Series 4000 supports the IETF AppleTalk MIB.

Optional Checksumming.
In order to be compatible with older Kinetics software, the checksum in the DDP packet header may not be generated. This can be enabled or disabled by the user.

Routing broadcast interval selectable on remote ports.
To prevent excessive broadcasting of routes and delay on the serial links, the user may select the routing broadcast interval on remote ports. It requires that the same interval be configured at the remote end.

AppleTalk over X.25 Support.
Remote AppleTalk networks can be connected over an X.25 link.